

Alibaba Cloud

Apsara Stack Enterprise

User Guide - Analytics and
Artificial Intelligence

Product Version: 2012, Internal: V3.13.0

Document Version: 20210621

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).
5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.
6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
 Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 Danger: Resetting will result in the loss of user configuration data.
 Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 Warning: Restarting will cause business interruption. About 10 minutes are required to restart an instance.
 Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	 Notice: If the weight is set to 0, the server no longer receives new requests.
 Note	A note indicates supplemental instructions, best practices, tips, and other content.	 Note: You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click Settings > Network > Set network type .
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click OK .
Courier font	Courier font is used for commands	Run the <code>cd /d C:/window</code> command to enter the Windows system folder.
<i>Italic</i>	Italic formatting is used for parameters and variables.	<code>bae log list --instanceid</code> <i>Instance_ID</i>
[] or [a b]	This format is used for an optional value, where only one item can be selected.	<code>ipconfig [-all -t]</code>
{ } or {a b}	This format is used for a required value, where only one item can be selected.	<code>switch {active stand}</code>

Table of Contents

1. MaxCompute	93
1.1. What is MaxCompute	93
1.2. Usage notes	94
1.3. Preparations	95
1.3.1. Log on to the Apsara Uni-manager Management Conso... ..	95
1.3.2. Create an Apsara Stack tenant account	96
1.3.3. Create a project	96
1.3.4. Create a quota group	98
1.4. Quick start	98
1.4.1. Overview	99
1.4.2. Configure the MaxCompute client	100
1.4.3. Add and delete a user	101
1.4.4. Grant and view permissions	102
1.4.4.1. Overview	102
1.4.4.2. ACL-based authorization	102
1.4.4.3. Policy-based access control	103
1.4.4.4. Query permissions	104
1.4.5. Create and authorize a role	105
1.4.6. Create or delete a table	105
1.4.6.1. Create a table	105
1.4.6.2. Query table information	106
1.4.6.3. Delete a table	106
1.4.7. Import or export data	107
1.4.8. Run SQL	107
1.4.8.1. Overview	107
1.4.8.2. SELECT	107

1.4.8.3. INSERT	108
1.4.8.4. JOIN	108
1.4.8.5. Other limits	108
1.4.9. Compile and use UDFs	109
1.4.9.1. Overview	109
1.4.9.2. UDF example	109
1.4.9.3. UDAF example	110
1.4.9.4. UDTF example	110
1.4.10. Write and run a MapReduce program	111
1.4.11. Write and run a Graph job	112
1.4.12. Use Logview to view job running information	113
1.4.13. Use Logview V2.0 to view job running information	117
1.5. Basic concepts and common commands	124
1.5.1. Terms	124
1.5.2. Common commands	130
1.5.2.1. Overview	130
1.5.2.2. Project operations	130
1.5.2.3. Table operations	132
1.5.2.4. Instance operations	136
1.5.2.5. Resource operations	140
1.5.2.6. Function operations	142
1.5.2.7. Time zone configuration operations	144
1.5.2.8. Tunnel operations	145
1.5.2.9. Other operations	155
1.6. MaxCompute SQL	159
1.6.1. Overview	159
1.6.1.1. Scenarios	159
1.6.1.2. Reserved words	159

1.6.1.3. Partitioned tables	159
1.6.1.4. Type conversion	160
1.6.1.4.1. Explicit conversions of data types	160
1.6.1.4.2. Implicit type conversion and its scope	161
1.6.1.4.3. SQL built-in functions	164
1.6.1.4.4. CASE WHEN	164
1.6.1.4.5. Conversions between the STRING and DATETIME...	164
1.6.2. Operators	165
1.6.2.1. Relational operators	165
1.6.2.2. Arithmetic operators	166
1.6.2.3. Bitwise operators	167
1.6.2.4. Logical operators	167
1.6.3. LOAD	168
1.6.4. DDL statements	171
1.6.4.1. Table operations	171
1.6.4.1.1. Create a table	171
1.6.4.1.2. Delete a table	176
1.6.4.1.3. Rename a table	177
1.6.4.1.4. Change the owner of a table	177
1.6.4.1.5. Modify the comment of a table	177
1.6.4.1.6. Modify the lifecycle settings of a table	178
1.6.4.1.7. Disable or enable the lifecycle feature	178
1.6.4.1.8. Modify the value of LastDataModifiedTime for a...	179
1.6.4.1.9. Modify the clustering attributes of a table	179
1.6.4.1.10. Delete data from a non-partitioned table	180
1.6.4.1.11. Archive table data	181
1.6.4.1.12. Forcibly delete data from a table or partition	182
1.6.4.2. View-based operation	182

1.6.4.2.1. Create a view	182
1.6.4.2.2. Delete a view	183
1.6.4.2.3. Rename a view	183
1.6.4.3. Column and partition operations	184
1.6.4.3.1. Add partitions	184
1.6.4.3.2. Delete a partition	184
1.6.4.3.3. Add columns or comments	187
1.6.4.3.4. Change the name of a column	187
1.6.4.3.5. Modify the comment of a column	188
1.6.4.3.6. Modify the name and comment of a column at ...	188
1.6.4.3.7. Modify the value of LastDataModifiedTime for a...	189
1.6.4.3.8. Modify partition values	189
1.6.4.3.9. Merge partitions	190
1.6.5. DML statements	191
1.6.5.1. INSERT	191
1.6.5.1.1. Update data in a table	191
1.6.5.1.2. Insert data into multiple objects	194
1.6.5.1.3. Insert data into dynamic partitions	195
1.6.5.1.4. VALUES	197
1.6.5.2. SELECT	200
1.6.5.2.1. SELECT operations	200
1.6.5.2.2. Subquery	206
1.6.5.2.3. UNION ALL	207
1.6.5.2.4. JOIN	207
1.6.5.2.5. MAPJOIN hint	209
1.6.5.2.6. SELECT TRANSFORM	211
1.6.5.2.6.1. Overview	211
1.6.5.2.6.2. SELECT TRANSFORM examples	212

1.6.5.2.6.3. Performance	216
1.6.5.2.7. GROUPING SETS	216
1.6.5.2.7.1. Overview	216
1.6.5.2.7.2. Example	216
1.6.5.2.7.3. CUBE and ROLLUP	217
1.6.5.2.7.4. GROUPING and GROUPING_ID	218
1.6.5.2.8. UNION, INTERSECT, and EXCEPT	219
1.6.5.3. EXPLAIN	222
1.6.5.4. IF statement	225
1.6.5.5. UPDATE and DELETE	226
1.6.6. Built-in functions	233
1.6.6.1. Mathematical functions	234
1.6.6.1.1. ABS	234
1.6.6.1.2. ACOS	234
1.6.6.1.3. ASIN	235
1.6.6.1.4. ATAN	235
1.6.6.1.5. CEIL	235
1.6.6.1.6. CONV	236
1.6.6.1.7. COS	236
1.6.6.1.8. COSH	237
1.6.6.1.9. COT	237
1.6.6.1.10. EXP	237
1.6.6.1.11. FLOOR	238
1.6.6.1.12. LN	238
1.6.6.1.13. LOG	239
1.6.6.1.14. POW	239
1.6.6.1.15. RAND	240
1.6.6.1.16. ROUND	240

1.6.6.1.17. SIN	241
1.6.6.1.18. SINH	241
1.6.6.1.19. SQRT	242
1.6.6.1.20. TAN	242
1.6.6.1.21. TANH	242
1.6.6.1.22. TRUNC	242
1.6.6.1.23. Additional mathematical functions	243
1.6.6.1.23.1. Usage notes	243
1.6.6.1.23.2. LOG2	244
1.6.6.1.23.3. LOG10	244
1.6.6.1.23.4. BIN	244
1.6.6.1.23.5. HEX	245
1.6.6.1.23.6. UNHEX	245
1.6.6.1.23.7. RADIANS	246
1.6.6.1.23.8. DEGREES	246
1.6.6.1.23.9. SIGN	246
1.6.6.1.23.10. E	247
1.6.6.1.23.11. PI	247
1.6.6.1.23.12. FACTORIAL	247
1.6.6.1.23.13. CBRT	248
1.6.6.1.23.14. SHIFLEFT	248
1.6.6.1.23.15. SHIFTRIGHT	248
1.6.6.1.23.16. SHIFTRIGHTUNSIGNED	249
1.6.6.1.23.17. FORMAT_NUMBER	249
1.6.6.1.23.18. WIDTH_BUCKET	250
1.6.6.2. String functions	251
1.6.6.2.1. CHAR_MATCHCOUNT	251
1.6.6.2.2. CHR	251

1.6.6.2.3. CONCAT	252
1.6.6.2.4. INSTR	252
1.6.6.2.5. IS_ENCODING	253
1.6.6.2.6. KEYVALUE	253
1.6.6.2.7. LENGTH	254
1.6.6.2.8. LENGTHB	255
1.6.6.2.9. MD5	255
1.6.6.2.10. PARSE_URL	255
1.6.6.2.11. REGEXP_EXTRACT	256
1.6.6.2.12. REGEXP_INSTR	257
1.6.6.2.13. REGEXP_SUBSTR	257
1.6.6.2.14. REGEXP_COUNT	258
1.6.6.2.15. REGEXP_REPLACE	258
1.6.6.2.16. SPLIT_PART	259
1.6.6.2.17. SUBSTR	260
1.6.6.2.18. TOLOWER	261
1.6.6.2.19. TOUPPER	261
1.6.6.2.20. TO_CHAR	261
1.6.6.2.21. TRIM	262
1.6.6.2.22. LTRIM	262
1.6.6.2.23. RTRIM	263
1.6.6.2.24. REVERSE	263
1.6.6.2.25. SPACE	264
1.6.6.2.26. REPEAT	264
1.6.6.2.27. ASCII	265
1.6.6.2.28. Additional string functions	265
1.6.6.2.28.1. Usage notes	265
1.6.6.2.28.2. CONCAT_WS	265

1.6.6.2.28.3. LPAD	266
1.6.6.2.28.4. RPAD	266
1.6.6.2.28.5. REPLACE	267
1.6.6.2.28.6. SOUNDEX	267
1.6.6.2.28.7. SUBSTRING_INDEX	267
1.6.6.2.28.8. TRANSLATE	268
1.6.6.2.28.9. URL_ENCODE	268
1.6.6.2.28.10. URL_DECODE	269
1.6.6.2.28.11. JSON_TUPLE	269
1.6.6.2.28.12. FROM_JSON	272
1.6.6.2.28.13. TO_JSON	273
1.6.6.3. Date functions	273
1.6.6.3.1. DATEADD	273
1.6.6.3.2. DATEDIFF	275
1.6.6.3.3. DATEPART	275
1.6.6.3.4. DATETRUNC	276
1.6.6.3.5. GETDATE	276
1.6.6.3.6. ISDATE	277
1.6.6.3.7. LASTDAY	277
1.6.6.3.8. TO_DATE	277
1.6.6.3.9. TO_CHAR	278
1.6.6.3.10. UNIX_TIMESTAMP	279
1.6.6.3.11. FROM_UNIXTIME	279
1.6.6.3.12. WEEKDAY	280
1.6.6.3.13. WEEKOFYEAR	280
1.6.6.3.14. Additional date functions	281
1.6.6.3.14.1. Usage notes	281
1.6.6.3.14.2. YEAR	281

1.6.6.3.14.3. QUARTER	282
1.6.6.3.14.4. MONTH	282
1.6.6.3.14.5. DAY	283
1.6.6.3.14.6. DAYOFMONTH	283
1.6.6.3.14.7. HOUR	283
1.6.6.3.14.8. MINUTE	284
1.6.6.3.14.9. SECOND	284
1.6.6.3.14.10. FROM_UTC_TIMESTAMP	285
1.6.6.3.14.11. CURRENT_TIMESTAMP	285
1.6.6.3.14.12. ADD_MONTHS	285
1.6.6.3.14.13. LAST_DAY	286
1.6.6.3.14.14. NEXT_DAY	286
1.6.6.3.14.15. MONTHS_BETWEEN	287
1.6.6.3.14.16. EXTRACT	287
1.6.6.4. Window functions	288
1.6.6.4.1. Overview	288
1.6.6.4.2. COUNT	289
1.6.6.4.3. AVG	291
1.6.6.4.4. MAX	292
1.6.6.4.5. MIN	293
1.6.6.4.6. MEDIAN	293
1.6.6.4.7. STDDEV	294
1.6.6.4.8. STDDEV_SAMP	295
1.6.6.4.9. SUM	295
1.6.6.4.10. DENSE_RANK	297
1.6.6.4.11. RANK	298
1.6.6.4.12. LAG	299
1.6.6.4.13. LEAD	300

1.6.6.4.14. PERCENT_RANK	301
1.6.6.4.15. ROW_NUMBER	301
1.6.6.4.16. CLUSTER_SAMPLE	303
1.6.6.4.17. NTILE	304
1.6.6.4.18. NTH_VALUE	305
1.6.6.4.19. CUME_DIST	307
1.6.6.4.20. FIRST_VALUE	308
1.6.6.4.21. LAST_VALUE	309
1.6.6.5. Aggregate functions	310
1.6.6.5.1. Expressions of filter conditions in aggregate fun...	311
1.6.6.5.2. COUNT	311
1.6.6.5.3. COUNT_IF	312
1.6.6.5.4. AVG	313
1.6.6.5.5. MAX	314
1.6.6.5.6. MIN	314
1.6.6.5.7. MEDIAN	315
1.6.6.5.8. STDDEV	316
1.6.6.5.9. STDDEV_SAMP	316
1.6.6.5.10. SUM	317
1.6.6.5.11. WM_CONCAT	318
1.6.6.5.12. PERCENTILE	319
1.6.6.5.13. Additional aggregate functions	319
1.6.6.5.13.1. Usage notes	319
1.6.6.5.13.2. COLLECT_LIST	320
1.6.6.5.13.3. COLLECT_SET	320
1.6.6.5.13.4. VARIANCE or VAR_POP	320
1.6.6.5.13.5. VAR_SAMP	321
1.6.6.5.13.6. COVAR_POP	322

1.6.6.5.13.7. COVAR_SAMP	322
1.6.6.5.13.8. ANY_VALUE	323
1.6.6.5.13.9. NUMERIC_HISTOGRAM	324
1.6.6.5.13.10. PERCENTILE_APPROX	324
1.6.6.5.13.11. APPROX_DISTINCT	324
1.6.6.6. Other functions	325
1.6.6.6.1. ARRAY	325
1.6.6.6.2. ARRAY_CONTAINS	325
1.6.6.6.3. CAST	326
1.6.6.6.4. COALESCE	326
1.6.6.6.5. DECODE	326
1.6.6.6.6. EXPLODE	327
1.6.6.6.7. GET_IDCARD_AGE	328
1.6.6.6.8. GET_IDCARD_BIRTHDAY	328
1.6.6.6.9. GET_IDCARD_SEX	329
1.6.6.6.10. GREATEST	329
1.6.6.6.11. INDEX	329
1.6.6.6.12. MAX_PT	330
1.6.6.6.13. ORDINAL	331
1.6.6.6.14. LEAST	332
1.6.6.6.15. SIZE	332
1.6.6.6.16. SPLIT	332
1.6.6.6.17. STR_TO_MAP	333
1.6.6.6.18. UUID	334
1.6.6.6.19. UNIQUE_ID	334
1.6.6.6.20. SAMPLE	334
1.6.6.6.21. CASE WHEN expression	335
1.6.6.6.22. IF	336

1.6.6.6.23. Additional functions	336
1.6.6.6.23.1. Usage notes	336
1.6.6.6.23.2. MAP	337
1.6.6.6.23.3. MAP_KEYS	337
1.6.6.6.23.4. MAP_VALUES	338
1.6.6.6.23.5. SORT_ARRAY	338
1.6.6.6.23.6. POSEXPLODE	339
1.6.6.6.23.7. STRUCT	339
1.6.6.6.23.8. NAMED_STRUCT	340
1.6.6.6.23.9. INLINE	340
1.6.6.6.23.10. BETWEEN AND expression	341
1.6.6.6.23.11. NVL	342
1.6.6.6.23.12. TABLE_EXISTS	342
1.6.6.6.23.13. PARTITION_EXISTS	343
1.6.7. UDFs	343
1.6.7.1. Overview	343
1.6.7.2. Types of parameters and return values	344
1.6.7.3. UDF	347
1.6.7.4. UDAF	348
1.6.7.5. UDTFs	351
1.6.7.5.1. Overview	351
1.6.7.5.2. Examples of using UDTFs	352
1.6.7.6. Python UDFs	356
1.6.7.6.1. Limits	356
1.6.7.6.2. Third-party libraries	357
1.6.7.6.3. Types of parameters and return values	357
1.6.7.6.4. UDF	359
1.6.7.6.5. UDAF	359

1.6.7.6.6. UDTF	360
1.6.7.6.7. Resource reference	361
1.6.7.7. Python 3 UDF	362
1.6.7.8. SQL Function	363
1.6.7.9. Embedded UDF	365
1.6.7.9.1. Background information	365
1.6.7.9.2. Feature summary	366
1.6.7.9.3. Limits	366
1.6.7.9.4. Examples	366
1.6.8. UDTs	368
1.6.8.1. Scenarios and limits	368
1.6.8.2. Feature summary	369
1.6.8.3. Implementation principles and feature description	370
1.6.8.4. Feature advantages	373
1.6.8.5. Performance advantages	374
1.6.8.6. Security advantages	374
1.6.8.7. More examples	374
1.6.8.7.1. Example of using Java arrays	374
1.6.8.7.2. Example of using JSON	375
1.6.8.7.3. Example of using complex data types	375
1.6.8.7.4. Aggregation example	376
1.6.8.7.5. Example of using table-valued functions	377
1.6.9. UDJ	377
1.6.9.1. Background information	377
1.6.9.2. Examples of using UDJ	378
1.6.9.2.1. Cross join operation by using UDJ	378
1.6.9.2.2. Pre-sorting	382
1.6.9.3. Performance	383

1.6.10. Parameterized view	384
1.6.11. CLONE TABLE	387
1.6.12. Geographic functions	389
1.6.12.1. Usage notes	389
1.6.12.2. Constructors	389
1.6.12.2.1. ST_AsBinary	389
1.6.12.2.2. ST_AsGeoJson	390
1.6.12.2.3. ST_AsJson	390
1.6.12.2.4. ST_AsShape	391
1.6.12.2.5. ST_AsText	391
1.6.12.2.6. ST_GeomCollection	391
1.6.12.2.7. ST_GeomFromGeoJson	392
1.6.12.2.8. ST_GeomFromJSON	392
1.6.12.2.9. ST_GeomFromShape	392
1.6.12.2.10. ST_GeomFromText	393
1.6.12.2.11. ST_GeomFromWKB	393
1.6.12.2.12. ST_GeometryType	393
1.6.12.2.13. ST_LineString	394
1.6.12.2.14. ST_LineFromWKB	394
1.6.12.2.15. ST_MultiLineString	395
1.6.12.2.16. ST_MLineFromWKB	395
1.6.12.2.17. ST_MultiPoint	395
1.6.12.2.18. ST_MPointFromWKB	395
1.6.12.2.19. ST_MultiPolygon	396
1.6.12.2.20. ST_MPolyFromWKB	396
1.6.12.2.21. ST_Point	396
1.6.12.2.22. ST_PointFromWKB	397
1.6.12.2.23. ST_PointZ	397

1.6.12.2.24. ST_Polygon	397
1.6.12.2.25. ST_PolyFromWKB	398
1.6.12.2.26. ST_SetSRID	398
1.6.12.3. Accessors	398
1.6.12.3.1. ST_Area	398
1.6.12.3.2. ST_Centroid	399
1.6.12.3.3. ST_CoordDim	399
1.6.12.3.4. ST_Dimension	400
1.6.12.3.5. ST_Distance	400
1.6.12.3.6. ST_GeodesicLengthWGS84	401
1.6.12.3.7. ST_GeometryN	401
1.6.12.3.8. ST_Is3D	402
1.6.12.3.9. ST_IsClosed	402
1.6.12.3.10. ST_IsEmpty	402
1.6.12.3.11. ST_IsMeasured	403
1.6.12.3.12. ST_IsSimple	403
1.6.12.3.13. ST_IsRing	403
1.6.12.3.14. ST_Length	404
1.6.12.3.15. ST_M	404
1.6.12.3.16. ST_MaxM	404
1.6.12.3.17. ST_MinM	405
1.6.12.3.18. ST_X	405
1.6.12.3.19. ST_Y	406
1.6.12.3.20. ST_Z	406
1.6.12.3.21. ST_MaxX	406
1.6.12.3.22. ST_MaxY	407
1.6.12.3.23. ST_MaxZ	407
1.6.12.3.24. ST_MinX	408

1.6.12.3.25. ST_MinY	408
1.6.12.3.26. ST_MinZ	409
1.6.12.3.27. ST_NumGeometries	409
1.6.12.3.28. ST_NumInteriorRing	410
1.6.12.3.29. ST_NumPoints	410
1.6.12.3.30. ST_PointN	411
1.6.12.3.31. ST_StartPoint	411
1.6.12.3.32. ST_EndPoint	411
1.6.12.3.33. ST_SRID	412
1.6.12.4. Operations	412
1.6.12.4.1. ST_Aggr_ConvexHull	412
1.6.12.4.2. ST_Aggr_Intersection	412
1.6.12.4.3. ST_Aggr_Union	413
1.6.12.4.4. ST_Bin	413
1.6.12.4.5. ST_BinEnvelope	413
1.6.12.4.6. ST_Boundary	413
1.6.12.4.7. ST_Buffer	414
1.6.12.4.8. ST_ConvexHull	414
1.6.12.4.9. ST_Difference	414
1.6.12.4.10. ST_Envelope	415
1.6.12.4.11. ST_ExteriorRing	415
1.6.12.4.12. ST_InteriorRingN	416
1.6.12.4.13. ST_Intersection	416
1.6.12.4.14. ST_SymmetricDiff	417
1.6.12.4.15. ST_Union	418
1.6.12.5. Relationship tests	418
1.6.12.5.1. ST_Contains	418
1.6.12.5.2. ST_Crosses	418

1.6.12.5.3. ST_Disjoint	419
1.6.12.5.4. ST_EnvIntersects	419
1.6.12.5.5. ST_Equals	419
1.6.12.5.6. ST_Intersects	420
1.6.12.5.7. ST_Overlaps	420
1.6.12.5.8. ST_Relate	420
1.6.12.5.9. ST_Touches	421
1.6.12.5.10. ST_Within	421
1.6.12.6. Geohash index functions	422
1.6.12.6.1. ST_GeoHash	422
1.6.12.6.2. ST_PointFromGeoHash	422
1.6.12.6.3. ST_EnvelopeFromGeoHash	422
1.6.12.6.4. ST_GeoHashNeighbours	423
1.6.12.7. S2 mesh functions	423
1.6.12.7.1. ST_S2CellIdsFromGeom	423
1.6.12.7.2. ST_S2CellIdsFromText	423
1.6.12.7.3. ST_S2CellCenterPoint	423
1.6.12.7.4. ST_S2CellNeighbours	424
1.6.12.8. Geodesic functions	424
1.6.12.8.1. ST_AreaWGS84	424
1.6.12.8.2. ST_DistanceWGS84	424
1.6.12.8.3. ST_BufferWGS84	425
1.6.12.8.4. ST_GeodesicDistance	425
1.6.12.8.5. ST_Distance_Sphere	425
1.6.12.8.6. ST_Area_Sphere	426
1.6.12.9. R-tree index functions	426
1.6.12.9.1. ST_BuildRTreeIndex	427
1.6.12.9.2. ST_ContainsFromRTree	427

1.6.12.9.3. ST_CrossesFromRTree	427
1.6.12.9.4. ST_EqualsFromRTree	427
1.6.12.9.5. ST_IntersectsFromRTree	428
1.6.12.9.6. ST_OverlapsFromRTree	428
1.6.12.9.7. ST_TouchesFromRTree	428
1.6.12.9.8. ST_WithinFromRTree	428
1.6.12.9.9. ST_KNNFromRTree	430
1.6.12.10. Other functions	430
1.6.12.10.1. ST_IsValid	430
1.6.12.10.2. ST_Transform	431
1.6.13. MaxCompute Hash Clustering	431
1.6.13.1. Background information	431
1.6.13.2. Descriptions	433
1.6.13.2.1. Enable Hash Clustering	433
1.6.13.2.2. Create a hash-clustered table	433
1.6.13.2.3. Modify table attributes	435
1.6.13.2.4. View and verify table attributes	435
1.6.13.3. Benefits	436
1.6.13.3.1. Bucket pruning and index optimization	436
1.6.13.3.2. Aggregation optimization	436
1.6.13.3.3. Storage optimization	437
1.6.13.4. ShuffleRemove	438
1.6.13.5. Limits	438
1.6.14. Common MaxCompute SQL parameter settings	439
1.6.14.1. Map configurations	439
1.6.14.2. Join configurations	439
1.6.14.3. Reduce configurations	440
1.6.14.4. UDF configurations	440

1.6.14.5. MapJoin configurations	440
1.6.14.6. Configurations of data skew	441
1.6.15. MapReduce-to-SQL conversion for execution	441
1.6.15.1. Overview	441
1.6.15.2. Configure the MaxCompute client	442
1.6.15.3. Configure running settings in DataWorks	442
1.6.15.4. View details	443
1.6.15.5. Perform operations on the distributed file system	445
1.6.16. Analysis of the mapping between SQL input and output	445
1.6.16.1. Overview	445
1.6.16.2. Usage notes	446
1.6.17. Common MaxCompute SQL errors and solutions	447
1.6.17.1. Data skew	447
1.6.17.1.1. Overview	447
1.6.17.1.2. GROUP BY data skew	448
1.6.17.1.3. DISTRIBUTE BY data skew	448
1.6.17.1.4. Data skew caused by JOIN operations	448
1.6.17.1.5. Data skew caused by multiple DISTINCT operations	448
1.6.17.1.6. Data skew caused by misuse of dynamic partitioning	449
1.6.17.1.7. Use SKEWJOIN HINT to avoid skewed hot key values	449
1.6.17.2. Insufficient computing resources	456
1.6.17.3. Methods to optimize storage in MaxCompute SQL	457
1.6.17.4. UDF OOM	459
1.6.18. Limits of MaxCompute SQL statements	459
1.6.19. Appendix	461
1.6.19.1. Escape characters	461
1.6.19.2. LIKE usage	462
1.6.19.3. Regular expressions	462

1.6.19.4. Reserved words and keywords	465
1.6.19.5. Settings of new data types	466
1.6.19.6. ACID semantics of MaxCompute concurrent write j...	466
1.7. MaxCompute Tunnel	468
1.7.1. Overview	468
1.7.2. Selection of tools to migrate data to the cloud	469
1.7.3. Introduction to the tools	470
1.7.4. Tunnel SDK overview	471
1.7.4.1. Overview	471
1.7.4.2. TableTunnel	472
1.7.4.3. InstanceTunnel	473
1.7.4.4. UploadSession	474
1.7.4.5. DownloadSession	476
1.7.4.6. TunnelBufferedWriter	477
1.7.5. Tunnel SDK example	478
1.7.5.1. Example of simple uploads	478
1.7.5.2. Example of simple downloads	479
1.7.5.3. Example of multi-thread uploads	481
1.7.5.4. Example of multi-thread downloads	483
1.7.5.5. Example of uploading data by using BufferedWriter	485
1.7.5.6. Example of uploading data by using BufferedWriter...	485
1.7.5.7. Examples of uploading and downloading data of co...	486
1.7.6. MaxCompute Streaming Tunnel	489
1.7.7. Appendix	491
1.7.7.1. FAQ related to data upload and download by using ...	491
1.7.7.2. Common tunnel error codes	493
1.8. MaxCompute MapReduce	495
1.8.1. Overview	495

1.8.1.1. MapReduce	495
1.8.1.2. MapReduce 2	497
1.8.1.3. Compatibility with Hadoop MapReduce	498
1.8.2. Limits	503
1.8.3. Features	504
1.8.3.1. Run command	504
1.8.3.2. Concepts	505
1.8.3.2.1. Map/Reduce	505
1.8.3.2.2. Sorting	505
1.8.3.2.3. Partition	505
1.8.3.2.4. Combiner	505
1.8.3.3. Submit a job	506
1.8.3.4. Inputs and outputs	507
1.8.3.5. Use resources	507
1.8.3.6. Job running in local mode	508
1.8.4. SDK introduction	510
1.8.4.1. Overview of major MapReduce APIs	510
1.8.4.2. API description	511
1.8.4.2.1. MapperBase	511
1.8.4.2.2. ReducerBase	511
1.8.4.2.3. TaskContext	511
1.8.4.2.4. JobConf	512
1.8.4.2.5. JobClient	514
1.8.4.2.6. RunningJob	514
1.8.4.2.7. InputUtils	514
1.8.4.2.8. OutputUtils	515
1.8.4.2.9. Pipeline	515
1.8.4.3. Compatibility with Hadoop MapReduce	516

1.8.5. Data types	530
1.8.6. Sample programs	530
1.8.6.1. WordCount example	530
1.8.6.2. MapOnly example	532
1.8.6.3. MultipleInOut example	534
1.8.6.4. MultiJobs example	536
1.8.6.5. SecondarySort example	538
1.8.6.6. Resource usage example	540
1.8.6.7. Counter usage example	542
1.8.6.8. Grep example	543
1.8.6.9. Join example	546
1.8.6.10. Sleep example	548
1.8.6.11. Unique example	549
1.8.6.12. Sort example	551
1.8.6.13. Examples of using partitioned tables as input	553
1.8.6.14. Pipeline example	554
1.9. MaxCompute Graph	556
1.9.1. Graph overview	556
1.9.1.1. Overview	556
1.9.1.2. Data structure of MaxCompute Graph	557
1.9.1.3. Graph logic	558
1.9.1.3.1. Graph loading	558
1.9.1.3.2. Iterative computing	558
1.9.1.3.3. Iteration termination	559
1.9.1.4. Aggregator mechanism	559
1.9.2. Graph feature overview	566
1.9.2.1. Run a job	566
1.9.2.2. Input and output	568

1.9.2.3. Read data from resources	569
1.9.2.3.1. Use GraphJob to specify resources to be read	569
1.9.2.3.2. Use resources in Graph	569
1.9.3. Graph SDK	569
1.9.4. Development and debugging	570
1.9.4.1. Development process	570
1.9.4.2. Development example	571
1.9.4.3. Perform debugging on the local computer	571
1.9.4.4. Temporary directory of a local job	573
1.9.4.5. Cluster debugging	574
1.9.4.6. Performance optimization	574
1.9.4.7. Built-in JAR packages	576
1.9.5. Limits	576
1.9.6. Sample programs	577
1.9.6.1. SSSP	577
1.9.6.2. PageRank	579
1.9.6.3. K-means clustering	581
1.9.6.4. BiPartiteMatchiing	585
1.9.6.5. Strongly connected components	588
1.9.6.6. Connected components	595
1.9.6.7. Topological sorting	597
1.9.6.8. Linear regression	600
1.9.6.9. Triangle count	604
1.9.6.10. Edge table import	606
1.9.6.11. Vertex table import	612
1.10. Java SDK	618
1.11. PyODPS	618
1.11.1. Quick start	618

1.11.2. Installation guide	619
1.11.3. Platform instructions	620
1.11.3.1. Overview	620
1.11.3.2. Use local PyODPS	620
1.11.3.3. Use PyODPS in DataWorks	621
1.11.4. Basic operations	623
1.11.4.1. Overview	623
1.11.4.2. Projects	623
1.11.4.3. Tables	623
1.11.4.4. SQL	630
1.11.4.5. Task instances	632
1.11.4.6. Resources	634
1.11.4.7. Functions	636
1.11.5. DataFrame	636
1.11.6. User experience enhancement	641
1.11.6.1. Command line	641
1.11.6.2. IPython	643
1.11.6.3. Jupyter Notebook	646
1.11.7. Configurations	649
1.11.8. API overview	652
1.11.9. FAQ	652
1.12. Java sandbox limits	654
1.13. Volume lifecycle management	654
1.13.1. Overview	654
1.13.2. Manage the lifecycle of a volume	654
1.14. Spark on MaxCompute	655
1.14.1. Overview	655
1.14.2. Project resources	655

1.14.3. Environment settings	655
1.14.3.1. Decompress the Spark on MaxCompute release pac...	655
1.14.3.2. Set environment variables	656
1.14.3.3. Configure Spark-defaults.conf	656
1.14.4. Quick start	657
1.14.5. Demo	659
1.14.6. Common cases	660
1.14.6.1. WordCount example	660
1.14.6.2. OSS access example	661
1.14.6.3. MaxCompute table read and write example	662
1.14.6.4. MaxCompute Table Spark-SQL example	664
1.14.6.5. Example of the self-developed client mode	665
1.14.6.6. MaxCompute Table PySpark example	666
1.14.6.7. MLLib example	666
1.14.6.8. PySpark interactive execution example	667
1.14.6.9. Spark-shell interactive execution example (read tab...	668
1.14.6.10. Spark-shell interactive execution example (Spark M..	668
1.14.6.11. Example of SparkR interactive execution	669
1.14.6.12. Example of PageRank with Apache Spark GraphX	669
1.14.6.13. Example of Spark Streaming-NetworkWordCount	671
1.14.7. Maven dependencies	671
1.14.8. Special notes	673
1.14.8.1. Running mode	673
1.14.8.2. Spark Streaming tasks	674
1.14.8.3. Job diagnosis	674
1.14.9. APIs supported by Spark	676
1.14.9.1. Spark Shell	676
1.14.9.2. Spark R	676

1.14.9.3. Spark SQL	677
1.14.9.4. Spark JDBC	677
1.14.10. Dynamic resource allocation of Spark	678
1.14.11. FAQ of Spark on MaxCompute	679
1.15. Elasticsearch on Maxcompute	680
1.15.1. Overview	680
1.15.2. Workflow	681
1.15.2.1. Overview	681
1.15.2.2. Distributed search workflow	681
1.15.2.3. Full-text index workflow	682
1.15.2.4. Authentication process	683
1.15.3. Quick start	683
1.15.4. Support for Elasticsearch applications	684
1.15.4.1. Typical practice of Elasticsearch on MaxCompute	684
1.15.4.2. Limits on Elasticsearch on MaxCompute in VPCs	684
1.15.5. Special notes	685
1.15.5.1. Find the Elasticsearch service domain name	685
1.15.5.2. Import table data from MaxCompute to Elasticsear...	685
1.16. Flink on MaxCompute	686
1.17. Non-structured data access and processing (integrated co...	688
1.17.1. Overview	688
1.17.2. Internal data sources	689
1.17.2.1. OSS data source	689
1.17.2.1.1. Overview	689
1.17.2.1.2. Use a built-in extractor to read data from OSS	689
1.17.2.1.2.1. Overview	689
1.17.2.1.2.2. Create an external table	689
1.17.2.1.2.3. Query an external table	690

1.17.2.1.2.4. MSCK REPAIR TABLE	691
1.17.2.1.2.5. Read data from the CSV or TSV files compr...	692
1.17.2.1.3. Use a custom extractor to read data from OSS	694
1.17.2.1.3.1. Overview	694
1.17.2.1.3.2. Define a storage handler	694
1.17.2.1.3.3. Define an extractor	695
1.17.2.1.3.4. Compile and package code	696
1.17.2.1.3.5. Create an external table	697
1.17.2.1.3.6. Query an external table	697
1.17.2.1.4. Use a custom extractor to read external unstruc...	698
1.17.2.1.5. Data partitions	700
1.17.2.1.5.1. Overview	700
1.17.2.1.5.2. Standard organization method and directory...	701
1.17.2.1.5.3. Custom directories of partition data in OSS	702
1.17.2.1.5.4. Access fully-customized non-partitioned data...	703
1.17.2.1.6. Output OSS data	703
1.17.2.1.6.1. Create an external table	703
1.17.2.1.6.2. Write data to a TSV text file by using an IN...	704
1.17.2.1.6.3. Write data to an unstructured file by using ...	705
1.17.2.1.6.4. Migrate data between different storage medi...	705
1.17.2.1.7. STS mode authorization for OSS	706
1.17.2.2. Tablestore data source	708
1.17.2.2.1. Overview	708
1.17.2.2.2. MaxCompute reads and computes data in Table...	709
1.17.2.2.2.1. Create an external table	709
1.17.2.2.2.2. Use an external table to access Tablestore d...	710
1.17.2.2.3. Write data from MaxCompute to Tablestore	710
1.17.2.3. AnalyticDB data source	711

1.17.2.3.1. Overview	711
1.17.2.3.2. Write data to AnalyticDB	711
1.17.2.3.2.1. Create an external table	711
1.17.2.3.2.2. Write and query data	712
1.17.2.3.3. Read data from AnalyticDB for MySQL	712
1.17.2.4. RDS data source	713
1.17.2.4.1. Overview	713
1.17.2.4.2. Write data to RDS	713
1.17.2.4.2.1. Create an external table	713
1.17.2.4.2.2. Write and query data	714
1.17.2.4.3. Read data from ApsaraDB RDS	714
1.17.2.5. HDFS data source (Alibaba Cloud)	714
1.17.2.5.1. Overview	714
1.17.2.5.2. Data processing for common tables	715
1.17.2.5.2.1. Write data to HDFS	715
1.17.2.5.2.2. Read data from Apsara File Storage for HD...	716
1.17.2.5.3. Data processing for partitioned tables	716
1.17.2.6. TDDL data source	717
1.17.2.6.1. Overview	717
1.17.2.6.2. Preparations	718
1.17.2.6.3. Create an external table	718
1.17.2.6.3.1. Syntax	718
1.17.2.6.3.2. Example	721
1.17.2.6.4. Read data from an external table	722
1.17.2.6.5. Write data to an external table	723
1.17.3. External data sources	723
1.17.3.1. HDFS data source (open-source)	724
1.17.3.1.1. Overview	724

1.17.3.1.2. Write data to HDFS	724
1.17.3.1.2.1. Create an external table	724
1.17.3.1.2.2. Write and query data	724
1.17.3.1.3. Read data from HDFS	725
1.17.3.2. MongoDB data source	725
1.17.3.2.1. Overview	725
1.17.3.2.2. Preparations	725
1.17.3.2.3. Write data to MongoDB	726
1.17.3.2.3.1. Create an external table	726
1.17.3.2.3.2. Write and query data	726
1.17.3.2.4. Read data from ApsaraDB for MongoDB	727
1.17.3.3. HBase data source	727
1.17.3.3.1. Overview	727
1.17.3.3.2. Write data to HBase	727
1.17.3.3.2.1. Write data to ApsaraDB for HBase by using ...	727
1.17.3.3.2.2. Write data to ApsaraDB for HBase by using...	728
1.17.3.3.3. Read data from ApsaraDB for HBase	731
1.18. Unstructured data access and processing (inside MaxCom... ..	731
1.18.1. Overview	731
1.18.2. Create a volume external table	732
1.18.2.1. Syntax	732
1.18.2.2. Use a built-in storage handler to create a table	733
1.18.2.3. Use a custom StorageHandler to create a table	733
1.18.3. Access a Volume external table	734
1.19. Multi-region cluster deployment on MaxCompute	734
1.19.1. Overview	734
1.19.2. Features of multi-region deployment	734
1.19.3. Instructions on multi-region deployment	735

1.19.4. Multi-region deployment examples	736
1.19.4.1. Synchronize table data among multiple clusters	736
1.19.4.2. Query the status of data synchronization between... ..	737
1.19.4.3. Cross-region direct read	738
1.19.4.4. Cross-region JOIN	740
1.20. Security solution	741
1.20.1. Intended users	741
1.20.2. Quick start	741
1.20.3. User authentication	744
1.20.4. Project user and authorization management	744
1.20.4.1. Overview	744
1.20.4.2. User management	744
1.20.4.3. Role management	745
1.20.4.4. ACL-based authorization actions	745
1.20.4.5. View permissions	748
1.20.5. Cross-project resource sharing	749
1.20.5.1. Overview	749
1.20.5.2. Package usage	750
1.20.5.2.1. Operations for package creators	750
1.20.5.2.2. Operations of package users	751
1.20.6. Project protection	752
1.20.6.1. Overview	752
1.20.6.2. Project data protection	752
1.20.6.3. Data export methods after project data protection... ..	753
1.20.6.4. Package-based resource sharing and project data	754
1.20.7. Project security configurations	755
1.20.8. Authorization policies	755
1.20.8.1. Policy overview	755

1.20.8.2. Policy-related terminology	757
1.20.8.3. Access policy structure	758
1.20.8.3.1. Overview	758
1.20.8.3.2. Authorization statement structure	759
1.20.8.3.3. Structure of condition blocks	759
1.20.8.3.4. Condition action types	759
1.20.8.3.5. Condition keys	760
1.20.8.4. Access policy norm	761
1.20.8.4.1. Principal naming conventions	761
1.20.8.4.2. Naming conventions of resources	761
1.20.8.4.3. Naming conventions of actions	762
1.20.8.4.4. Naming conventions of condition keys	763
1.20.8.4.5. Example of a policy for policy-based authoriza...	763
1.20.8.5. Differences between policy-based authorization an...	764
1.20.8.6. Limits	764
1.20.9. Collection of security statements	765
1.20.9.1. Project security configurations	765
1.20.9.2. Project permission management	766
1.20.9.3. Package-based resource sharing	767
1.21. Hierarchical throttling	768
1.21.1. Overview	768
1.21.2. Hierarchical throttling	768
1.21.3. Specify the maximum number of instances that are al...	769
1.22. Frequently-used tools	771
1.22.1. MaxCompute console	771
1.22.1.1. Usage notes	771
1.22.1.2. Install the MaxCompute client	771
1.22.1.3. Configuration-related operations	772

1.22.2. Eclipse development plugin	775
1.22.2.1. Install the Eclipse development plug-in	775
1.22.2.2. Create a project	777
1.22.2.2.1. Method 1	777
1.22.2.2.2. Method 2	780
1.22.2.3. MapReduce running example	781
1.22.2.3.1. Run a WordCount program	781
1.22.2.3.2. Run a custom MapReduce program	784
1.22.2.4. UDF development and running example	797
1.22.2.4.1. Local debug UDF programs	797
1.22.2.4.1.1. Run a UDF from the menu bar	797
1.22.2.4.1.2. Use the right-click shortcut menu to run a ...	798
1.22.2.4.2. Run a UDF	800
1.22.2.5. Graph operation example	802
1.23. MaxCompute feature enhancement packages	805
1.23.1. Content moderation	805
1.23.2. MCQA	807
1.23.2.1. Overview	807
1.23.2.2. Usage notes	809
1.23.3. VVP On MaxCompute	816
1.23.3.1. Overview	816
1.23.3.2. Benefits	816
1.23.3.3. Activate VVP On MaxCompute	817
1.23.3.4. Usage notes	818
1.24. MaxCompute FAQ	823
1.25. Open source features of MaxCompute	829
2. DataWorks	831
2.1. Log on to the DataWorks console	831

2.2. Create a workspace	832
2.3. Quick Start	832
2.3.1. Overview	832
2.3.2. Create tables and import data	833
2.3.3. Create a workflow	836
2.3.4. Create a sync node	838
2.3.5. Configure recurrence and dependencies for a node	840
2.3.6. Run a node and troubleshoot errors	842
2.4. Data Integration	844
2.4.1. Overview	844
2.4.2. Homepage	846
2.4.3. Connectivity testing	847
2.4.4. Data sources	848
2.4.4.1. Supported data stores and plug-ins	848
2.4.4.2. Connection isolation	849
2.4.4.3. Sync data monitoring	850
2.4.4.4. Manage connection permissions	850
2.4.4.5. Configure a MySQL connection	853
2.4.4.6. Configure an SQL Server data source	855
2.4.4.7. Configure a PostgreSQL connection	857
2.4.4.8. Configure an Oracle connection	858
2.4.4.9. Configure a Dameng connection	859
2.4.4.10. Configure a DRDS connection	860
2.4.4.11. Configure a PolarDB connection	862
2.4.4.12. Configure a HybridDB for MySQL connection	863
2.4.4.13. Configure a HybridDB for PostgreSQL connection	864
2.4.4.14. Configure an ApsaraDB for OceanBase connection	865
2.4.4.15. Configure a MaxCompute connection	866

2.4.4.16. Configure a DataHub connection	866
2.4.4.17. Configure an AnalyticDB for MySQL connection	867
2.4.4.18. Configure a Vertica connection	868
2.4.4.19. Configure a GBase8a connection	869
2.4.4.20. Configure a Lightning connection	870
2.4.4.21. Configure an HBase connection	871
2.4.4.22. Configure a Hologres connection	872
2.4.4.23. Configure a Hive connection	873
2.4.4.24. Configure an OSS connection	874
2.4.4.25. Configure an HDFS connection	876
2.4.4.26. Configure an FTP connection	876
2.4.4.27. Configure a MongoDB connection	877
2.4.4.28. Configure a Memcache connection	879
2.4.4.29. Configure a Redis connection	880
2.4.4.30. Configure a Tablestore connection	881
2.4.4.31. Configure an Elasticsearch connection	882
2.4.4.32. Configure a LogHub connection	883
2.4.5. Configure data synchronization tasks	884
2.4.5.1. Configure a sync node by using the codeless UI	884
2.4.5.2. Configure a sync node by using the code editor	887
2.4.5.3. Configure the reader	891
2.4.5.3.1. Configure DRDS Reader	891
2.4.5.3.2. Configure HBase Reader	896
2.4.5.3.3. Configure HDFS Reader	901
2.4.5.3.4. Configure MaxCompute Reader	910
2.4.5.3.5. Configure MongoDB Reader	914
2.4.5.3.6. Configure Db2 Reader	919
2.4.5.3.7. Configure MySQL Reader	923

2.4.5.3.8. Configure Oracle Reader	930
2.4.5.3.9. Configure OSS Reader	936
2.4.5.3.10. Configure FTP Reader	941
2.4.5.3.11. Configure Tablestore Reader	947
2.4.5.3.12. Configure PostgreSQL Reader	952
2.4.5.3.13. Configure SQL Server Reader	958
2.4.5.3.14. Configure LogHub Reader	965
2.4.5.3.15. Configure Tablestore Reader-Internal	970
2.4.5.3.16. Configure OTSStream Reader	976
2.4.5.3.17. Configure RDBMS Reader	981
2.4.5.3.18. Configure Stream Reader	987
2.4.5.3.19. Configure Hive Reader	989
2.4.5.3.20. Configure Elasticsearch Reader	990
2.4.5.3.21. Configure Vertica Reader	993
2.4.5.3.22. Configure GBase Reader	997
2.4.5.4. Configure the writer	1000
2.4.5.4.1. Configure AnalyticDB for MySQL 2.0 Writer	1001
2.4.5.4.2. Configure DataHub Writer	1004
2.4.5.4.3. Configure the DB2 writer	1007
2.4.5.4.4. Configure DRDS Writer	1010
2.4.5.4.5. Configure the FTP writer	1014
2.4.5.4.6. Configure HBase Writer	1018
2.4.5.4.7. Configure HBase11xsql Writer	1025
2.4.5.4.8. Configure HDFS Writer	1028
2.4.5.4.9. Configure MaxCompute Writer	1035
2.4.5.4.10. Configure Memcache Writer	1041
2.4.5.4.11. Configure MongoDB Writer	1044
2.4.5.4.12. Configure MySQL Writer	1049

2.4.5.4.13. Configure Oracle Writer	1053
2.4.5.4.14. Configure OSS Writer	1057
2.4.5.4.15. Configure PostgreSQL Writer	1062
2.4.5.4.16. Configure Redis Writer	1067
2.4.5.4.17. Configure SQL Server Writer	1072
2.4.5.4.18. Configure Elasticsearch Writer	1076
2.4.5.4.19. Configure LogHub Writer	1083
2.4.5.4.20. Configure Open Search Writer	1085
2.4.5.4.21. Configure Tablestore Writer	1089
2.4.5.4.22. Configure RDBMS Writer	1093
2.4.5.4.23. Configure Stream Writer	1096
2.4.5.4.24. Configure Hive Writer	1097
2.4.5.4.25. Configure Vertica Writer	1102
2.4.5.4.26. Configure Gbase8a Writer	1105
2.4.5.5. Optimize synchronization performance	1107
2.4.6. Real-time synchronization	1110
2.4.6.1. Overview	1110
2.4.6.2. Data sources supported by real-time synchronizatio...	1112
2.4.6.3. Create, configure, commit, and manage real-time s...	1113
2.4.6.4. Reader	1118
2.4.6.4.1. MySQL binlogs	1118
2.4.6.4.2. Oracle CDC	1121
2.4.6.4.3. DataHub	1123
2.4.6.4.4. LogHub	1125
2.4.6.4.5. Kafka	1127
2.4.6.4.6. Configure PolarDB Reader	1128
2.4.6.5. Writer	1129
2.4.6.5.1. Configure MaxCompute Writer	1129

2.4.6.5.2. Configure Hologres Writer	1130
2.4.6.5.3. Configure DataHub Writer	1132
2.4.6.5.4. Kafka	1133
2.4.6.6. Transform	1134
2.4.6.6.1. Data filter	1134
2.4.6.6.2. String replacement	1136
2.4.7. Data synchronization solutions	1136
2.4.7.1. Go to the Sync Solutions page	1136
2.4.7.2. Synchronize data to Hologres in real time	1137
2.4.7.3. Synchronize data to MaxCompute in real time	1139
2.4.8. Resource groups	1142
2.4.8.1. Overview	1142
2.4.8.2. Shared resource groups	1143
2.4.9. Full-database migration	1143
2.4.9.1. Overview	1143
2.4.9.2. Migrate a MySQL database	1144
2.4.9.3. Migrate Oracle databases	1146
2.5. Data Analytics	1147
2.5.1. Solution	1147
2.5.2. SQL coding guidelines and specifications	1148
2.5.3. GUI elements	1152
2.5.3.1. Overview	1152
2.5.3.2. Workflow Parameters	1154
2.5.3.3. Lineage	1157
2.5.3.4. Versions	1158
2.5.3.5. Code Structure	1160
2.5.4. Business flows	1164
2.5.4.1. Overview	1165

2.5.4.2. Create and reference a node group	1167
2.5.5. Node types	1169
2.5.5.1. Data Integration	1169
2.5.5.1.1. Create a batch sync node	1169
2.5.5.2. MaxCompute	1170
2.5.5.2.1. Create an ODPS SQL node	1170
2.5.5.2.2. Create an SQL Snippet node	1174
2.5.5.2.3. Create an ODPS Spark node	1175
2.5.5.2.4. Create a PyODPS node	1177
2.5.5.2.5. Create an ODPS Script node	1179
2.5.5.2.6. Create an ODPS MR node	1181
2.5.5.2.7. Create a MaxCompute table	1184
2.5.5.2.8. Create, reference, and download resources	1187
2.5.5.2.9. Register a UDF	1189
2.5.5.3. EMR	1190
2.5.5.3.1. Create an EMR MR node	1190
2.5.5.3.2. Create an EMR Spark SQL node	1191
2.5.5.3.3. Create an EMR Spark node	1192
2.5.5.3.4. Create an EMR Hive node	1193
2.5.5.4. Hologres	1193
2.5.5.4.1. Create a Hologres SQL node	1193
2.5.5.5. AnalyticDB for PostgreSQL	1194
2.5.5.5.1. Create an AnalyticDB for PostgreSQL node	1195
2.5.5.5.2. Create an AnalyticDB for PostgreSQL table	1196
2.5.5.6. Algorithm	1199
2.5.5.6.1. Create a PAI node	1199
2.5.5.7. General	1199
2.5.5.7.1. Create a for-each node	1199

2.5.5.7.2. Create a do-while node	1201
2.5.5.7.3. Create a merge node	1205
2.5.5.7.4. Create a branch node	1206
2.5.5.7.5. Create an assignment node	1208
2.5.5.7.6. Create a Shell node	1211
2.5.5.7.7. Create a zero-load node	1212
2.5.5.7.8. Create a cross-tenant collaboration node	1213
2.5.5.7.9. Create a data analysis report node	1214
2.5.5.7.10. Create a real-time sync node check node	1215
2.5.5.8. Custom	1216
2.5.5.8.1. Create a Hologres development node	1216
2.5.6. Schedule	1217
2.5.6.1. Basic properties	1217
2.5.6.2. Scheduling parameters	1218
2.5.6.3. Scheduling properties	1226
2.5.6.4. Dependencies	1234
2.5.7. Components	1236
2.5.7.1. Create a script template	1236
2.5.7.2. Use a script template	1242
2.5.8. Custom node type	1242
2.5.8.1. Overview	1242
2.5.8.2. Create a custom wrapper	1244
2.5.8.3. Create a custom node type	1246
2.5.9. Manage configurations	1247
2.5.9.1. Setup	1247
2.5.9.2. Configuration center	1248
2.5.9.3. Workspace settings	1251
2.5.9.4. Template management	1253

2.5.9.5. Folder management	1254
2.5.9.6. Level management	1254
2.5.9.7. Workspace backup and restore	1255
2.5.10. Deploy	1257
2.5.10.1. Deploy nodes	1258
2.5.10.2. Overview of cross-workspace cloning	1259
2.5.10.3. Clone nodes across workspaces	1260
2.5.11. Create an ad hoc query node	1260
2.5.12. View runtime logs	1261
2.5.13. View tenant tables	1262
2.5.14. Manage tables	1264
2.5.15. View built-in functions	1265
2.5.16. Manage deleted nodes	1265
2.5.17. Create a manually triggered workflow	1266
2.5.18. Editor keyboard shortcuts	1269
2.5.19. Use E-MapReduce in DataWorks	1271
2.6. HoloStudio	1272
2.6.1. Overview	1272
2.6.2. Bind a Hologres database to the current workspace	1273
2.6.3. SQL Console	1274
2.6.4. PostgreSQL management	1276
2.6.4.1. Manage databases	1276
2.6.4.2. Manage tables	1277
2.6.4.3. Manage foreign tables	1279
2.6.5. Data analytics	1280
2.6.5.1. Overview	1280
2.6.5.2. Use the Interactive Analytics Development submodu...	1280
2.6.5.3. Create multiple foreign tables at a time	1285

2.6.5.4. Import MaxCompute data	1286
2.6.5.5. Upload local files	1287
2.6.6. Hologres console	1289
2.6.6.1. Overview	1289
2.6.6.2. View the instance list	1289
2.6.6.3. Manage instances	1290
2.6.6.4. Manage users	1291
2.6.6.5. Manage databases	1292
2.7. Realtime Analysis	1293
2.7.1. Overview	1293
2.7.2. Workbook	1293
2.7.2.1. Create a workbook	1294
2.7.2.2. Edit a workbook	1294
2.7.3. Dimension tables	1303
2.7.3.1. Create and manage dimension tables	1303
2.7.3.2. Import data to a dimension table	1304
2.7.3.3. Edit a dimension table	1306
2.7.3.4. Share a dimension table	1307
2.7.4. Report	1308
2.7.4.1. Create a report	1308
2.7.4.2. Edit a report	1308
2.7.5. Go to the report center	1313
2.7.6. Go to the learning center	1314
2.8. Administration	1314
2.8.1. Overview	1314
2.8.2. Dashboard	1314
2.8.3. Manage real-time synchronization nodes	1317
2.8.4. Auto triggered node O&M	1319

2.8.4.1. Manage auto triggered nodes	1319
2.8.4.2. Manage auto triggered node instances	1321
2.8.4.3. Manage retroactive instances	1323
2.8.4.4. Manage test instances	1328
2.8.5. Manually triggered node O&M	1330
2.8.5.1. Manage manually triggered nodes	1330
2.8.5.2. Manage manually triggered node instances	1331
2.8.6. MaxCompute engine O&M	1332
2.8.7. Monitor	1333
2.8.7.1. Overview	1333
2.8.7.2. Feature description	1333
2.8.7.2.1. Baseline alert and event alert	1333
2.8.7.2.2. Custom alert trigger	1335
2.8.7.3. Instructions	1336
2.8.7.3.1. Baseline instances	1336
2.8.7.3.2. Baselines	1337
2.8.7.3.3. Events	1338
2.8.7.3.4. Alert triggers	1339
2.8.7.3.5. Alert information	1340
2.8.7.4. FAQ	1340
2.9. Security Center	1342
2.9.1. Overview	1342
2.9.2. My Permissions	1342
2.9.3. Authorizations	1344
2.9.4. Approval Center	1344
2.9.5. FAQ	1345
2.10. Data Quality	1347
2.10.1. Overview	1347

2.10.2. Features	1347
2.10.2.1. Dashboard	1347
2.10.2.2. My Subscriptions	1348
2.10.2.3. Configure monitoring rules	1348
2.10.2.4. View monitoring results	1352
2.10.2.5. Report Template Management	1354
2.10.2.6. Manage rule templates	1356
2.10.3. User guide	1362
2.10.3.1. Configure monitoring rules for MaxCompute	1362
2.10.3.2. Configure monitoring rules for DataHub	1368
2.11. Data Map	1371
2.11.1. Overview	1371
2.11.2. View overall data	1372
2.11.3. View and manage data and data permissions	1373
2.11.4. Manage categories and configure workspace permissions	1378
2.11.5. View table details	1380
2.11.5.1. View the details of a table	1380
2.11.5.2. Request table permissions	1383
2.11.5.3. Add a table to favorites	1384
2.11.6. Data discovery	1385
2.11.6.1. Collect metadata from an E-MapReduce data source	1385
2.11.6.2. Collect metadata from a MySQL data source	1385
2.11.6.3. Collect metadata from a PostgreSQL data source	1387
2.11.6.4. Collect metadata from an SQL Server data source	1388
2.11.6.5. Collect metadata from an Oracle data source	1389
2.11.6.6. Collect metadata from an AnalyticDB for PostgreSQL	1390
2.11.6.7. Collect metadata from an AnalyticDB for MySQL 2.x	1391
2.11.6.8. Collect metadata from an AnalyticDB for MySQL 3.x	1392

2.11.6.9. Collect metadata from a Hologres data source	1393
2.12. Data Asset Management	1394
2.12.1. Go to the Data Asset Management page	1394
2.12.2. Asset manager	1394
2.12.3. Asset user	1395
2.12.4. Asset administrator	1395
2.12.5. Manage authorizations	1399
2.12.6. Perform cross-tenant authorization	1399
2.13. Organization management	1401
2.13.1. Member management	1401
2.13.2. Resource groups	1401
2.13.2.1. Go to the page for managing scheduling resources	1401
2.13.2.2. Change the workspace to which a scheduling reso...	1401
2.13.3. Configure the compute engine	1402
2.14. Data Service	1402
2.14.1. Overview	1402
2.14.2. Terms	1403
2.14.3. Manage tags	1403
2.14.4. Manage business processes and objects under busines...	1405
2.14.4.1. Manage business processes	1405
2.14.4.2. Manage APIs	1408
2.14.4.3. Manage functions	1410
2.14.4.4. Manage workflows	1413
2.14.5. Create an API	1415
2.14.5.1. Configure connections	1416
2.14.5.2. Create an API in the codeless UI	1417
2.14.5.3. Create an API in the code editor	1424
2.14.5.4. Use filters	1429

2.14.5.4.1. Use prefilters	1429
2.14.5.4.2. Use post filters	1432
2.14.6. Register APIs	1435
2.14.7. Test APIs	1437
2.14.8. Publish APIs	1437
2.14.9. Call APIs	1438
2.14.10. Use workflows	1438
2.14.11. Manage versions	1444
2.14.12. FAQ	1445
2.15. Stream Studio	1446
2.15.1. Overview	1446
2.15.2. Bind a Realtime Compute project	1446
2.15.3. Create a real-time computing node	1446
2.15.4. Get started with Stream Studio	1447
2.15.5. Configure components	1451
2.15.5.1. Source tables	1451
2.15.5.1.1. Datahub	1451
2.15.5.1.2. Log Service	1453
2.15.5.2. Dimension tables	1455
2.15.5.2.1. ApsaraDB RDS	1455
2.15.5.2.2. Tablestore	1457
2.15.5.2.3. MaxCompute	1458
2.15.5.3. Data operators	1461
2.15.5.3.1. Filter	1461
2.15.5.3.2. GroupBy	1461
2.15.5.3.3. Join	1461
2.15.5.3.4. Select	1461
2.15.5.3.5. UDTF	1462

2.15.5.3.6. UnionAll	1462
2.15.5.3.7. Dynamic column splitting	1462
2.15.5.3.8. Static column splitting	1463
2.15.5.3.9. Row splitting	1463
2.15.5.4. Result tables	1464
2.15.5.4.1. Datahub	1464
2.15.5.4.2. Log Service	1465
2.15.5.4.3. ApsaraDB RDS	1466
2.15.5.4.4. Tablestore	1470
2.15.5.4.5. MaxCompute	1472
2.15.5.5. FAQ	1474
2.16. Data Protection	1475
2.16.1. Overview	1475
2.16.2. Configure rules for defining sensitive data	1475
2.16.3. View the distribution of sensitive data	1477
2.16.4. View the information about data activities	1477
2.16.5. View the data audited as risky	1477
2.16.6. Track data	1478
2.16.7. Manage a self-generated data recognition model	1479
2.16.8. Manage the data security levels	1481
2.16.9. Manage data that is incorrectly detected	1482
2.16.10. Customize de-identification rules	1482
2.16.11. Manage user groups	1484
2.17. App Studio	1485
2.17.1. Overview	1485
2.17.2. Get started with App Studio	1486
2.17.3. Navigation pane	1492
2.17.3.1. View and manage projects	1493

2.17.3.2. View and manage templates -----	1493
2.17.4. Project management -----	1493
2.17.5. Code editing -----	1494
2.17.5.1. Overview -----	1494
2.17.5.2. Generate code snippets -----	1495
2.17.5.3. Run UT -----	1496
2.17.5.4. Find in Path -----	1496
2.17.6. Debugging -----	1496
2.17.6.1. Configuration and startup -----	1496
2.17.6.2. Online debugging -----	1497
2.17.6.3. Breakpoint types -----	1498
2.17.6.4. Breakpoint operations -----	1499
2.17.6.5. Terminal -----	1500
2.17.6.6. Hot code replacement -----	1500
2.17.7. WYSIWYG designer -----	1501
2.17.7.1. Get started with the WYSIWYG designer -----	1501
2.17.7.2. Code mode -----	1502
2.17.7.3. DSL syntax -----	1503
2.17.7.4. Global data flow -----	1504
2.17.7.5. Save, preview, run, and hot code replacement -----	1506
2.17.7.6. Navigation configuration -----	1506
2.18. Migration Assistant -----	1507
2.18.1. Overview -----	1507
2.18.2. Cloud tasks -----	1508
2.18.2.1. Export tasks from open source engines -----	1508
2.18.2.2. Import tasks of open source engines -----	1510
2.18.3. Migrate data objects in DataWorks -----	1511
2.18.3.1. Create and view export tasks -----	1512

2.18.3.2. Create and view import tasks	1515
2.19. Workspace management	1517
2.19.1. Configure a workspace	1517
2.19.2. Manage workspace members	1521
2.19.3. Permission list	1522
2.19.4. Manage connections	1533
3.Realtime Compute	1534
3.1. What is Realtime Compute?	1534
3.2. Quick start	1535
3.2.1. Log on to the Realtime Compute console	1535
3.2.2. Real-time security monitoring	1535
3.2.2.1. Overview	1535
3.2.2.2. Preparations	1535
3.2.2.3. Develop a job	1537
3.2.2.4. Administration	1539
3.2.3. Frequently used words	1540
3.2.3.1. Overview	1540
3.2.3.2. Code development	1540
3.2.3.3. Code debugging	1541
3.2.3.4. Administration	1543
3.2.4. Big screen service for the Tmall Double Eleven Global	1543
3.2.4.1. Overview	1543
3.2.4.2. Scenario description	1544
3.2.4.3. Preparations	1544
3.2.4.4. Register a data store	1545
3.2.4.5. Development	1545
3.2.4.6. Administration	1546
3.3. Project management	1547

3.4. Data storage	1549
3.4.1. Overview	1549
3.4.2. Overview	1549
3.4.2.1. Overview	1549
3.4.2.2. Types	1550
3.4.2.3. Registration and usage	1550
3.4.3. Register a DataHub data store	1553
3.4.4. Register a Log Service data store	1554
3.4.5. Register a Tablestore data store	1556
3.4.6. Register an RDS data store	1556
3.5. Data development	1562
3.5.1. Create a job	1562
3.5.2. Development	1563
3.5.2.1. SQL code assistance	1563
3.5.2.2. SQL code version management	1563
3.5.2.3. Data store management	1564
3.5.3. Debug job code	1564
3.5.4. Publish a job SQL file	1566
3.5.5. View logs	1567
4. Machine Learning Platform for AI	1569
4.1. What is machine learning?	1569
4.2. Quick start	1569
4.2.1. Overview	1569
4.2.2. Log on to the Machine Learning Platform for AI console	1570
4.2.3. Data preparation	1571
4.2.4. Data preprocessing	1572
4.2.5. Data visualization	1573
4.2.6. Algorithm modeling	1573

4.2.7. Model prediction evaluation	1573
4.2.8. DataWorks task scheduling	1574
4.3. Online model service (must be activated separately)	1575
4.3.1. Deploy an online model service	1575
4.3.2. Create a service	1575
4.3.3. Add an existing service version	1576
4.3.4. Create a blue-green deployment	1576
4.4. Components	1576
4.4.1. Overview	1576
4.4.2. Data source and target	1577
4.4.3. Data preprocessing	1577
4.4.3.1. Sampling and filtering	1577
4.4.3.1.1. Random sampling	1577
4.4.3.1.2. Weighted sampling	1579
4.4.3.1.3. Filtering and mapping	1580
4.4.3.1.4. Stratified sampling	1581
4.4.3.2. Data merge	1583
4.4.3.2.1. Join	1583
4.4.3.2.2. Merge columns	1583
4.4.3.2.3. Merge rows (UNION)	1584
4.4.3.3. Others	1586
4.4.3.3.1. Add ID column	1586
4.4.3.3.2. Split	1587
4.4.3.3.3. Missing value imputation	1588
4.4.3.3.4. Normalization	1593
4.4.3.3.5. Standardization	1594
4.4.3.3.6. KV to Table	1600
4.4.3.3.7. Table to KV	1603

4.4.4. Feature engineering	1606
4.4.4.1. Feature transformation	1606
4.4.4.1.1. PCA	1606
4.4.4.2. Feature importance evaluation	1607
4.4.4.2.1. Linear model feature importance	1607
4.4.4.2.2. Random forest feature importance	1608
4.4.5. Statistical analysis	1609
4.4.5.1. Data pivoting	1610
4.4.5.2. Whole table statistics	1611
4.4.5.3. Correlation coefficient matrix	1611
4.4.5.4. Covariance	1614
4.4.5.5. Empirical probability density chart	1615
4.4.5.6. Chi-square goodness of fit test	1619
4.4.5.7. Chi-square test of independence	1621
4.4.5.8. Scatter plot	1623
4.4.5.9. Two-sample T-test	1628
4.4.5.10. One-sample T-test	1631
4.4.5.11. Lorenz curve	1632
4.4.5.12. Normality test	1636
4.4.5.13. Percentile	1638
4.4.5.14. Pearson coefficient	1639
4.4.5.15. Histogram	1640
4.4.6. Machine learning	1640
4.4.6.1. Binary classification	1640
4.4.6.1.1. GBDT binary classification	1640
4.4.6.1.2. Linear SVM	1643
4.4.6.1.3. Logistic regression for binary classification	1645
4.4.6.1.4. PS-SMART binary classification	1646

4.4.6.2. Multiclass classification	1654
4.4.6.2.1. KNN	1654
4.4.6.2.2. Logistic regression for multiclass classification	1657
4.4.6.2.3. Random forest	1658
4.4.6.2.4. Naive Bayes	1661
4.4.6.2.5. PS-SMART multiclass classification	1662
4.4.6.3. K-means clustering	1671
4.4.6.4. Regression	1673
4.4.6.4.1. GBDT regression	1673
4.4.6.4.2. Linear regression	1678
4.4.6.4.3. PS linear regression	1683
4.4.6.4.4. PS-SMART regression	1688
4.4.6.5. Collaborative filtering (etrec)	1697
4.4.6.6. Evaluation	1700
4.4.6.6.1. Regression model evaluation	1700
4.4.6.6.2. Clustering model evaluation	1701
4.4.6.6.3. Binary classification evaluation	1704
4.4.6.6.4. Confusion matrix	1705
4.4.6.6.5. Multiclass classification evaluation	1706
4.4.6.7. Prediction	1707
4.4.7. Deep learning (must be activated separately)	1709
4.4.7.1. Activate deep learning	1709
4.4.7.2. Read OSS buckets	1709
4.4.7.3. TensorFlow 1.4	1710
4.4.8. Time series	1713
4.4.8.1. x13_arima	1713
4.4.8.2. x13_auto_arima	1719
4.4.9. Text analysis	1725

4.4.9.1. Word splitting	1725
4.4.9.2. Deprecated word filtering	1728
4.4.9.3. String similarity	1729
4.4.9.4. Convert row, column, and value to KV pair	1732
4.4.9.5. String similarity - Top N	1734
4.4.9.6. N-gram counting	1737
4.4.9.7. Text summarization	1739
4.4.9.8. Keyword extraction	1741
4.4.9.9. Sentence splitting	1745
4.4.9.10. Semantic vector distance	1747
4.4.9.11. Document similarity	1748
4.4.9.12. PMI	1751
4.4.9.13. Word frequency statistics	1754
4.4.9.14. TF-IDF	1756
4.4.9.15. PLDA	1757
4.4.9.16. Word2Vec	1760
4.4.10. Network analysis	1762
4.4.10.1. K-core	1762
4.4.10.2. Single-source shortest path	1764
4.4.10.3. PageRank	1766
4.4.10.4. Label propagation clustering	1769
4.4.10.5. Label propagation classification	1772
4.4.10.6. Modularity	1775
4.4.10.7. Maximum connected subgraph	1777
4.4.10.8. Vertex clustering coefficient	1779
4.4.10.9. Edge clustering coefficient	1781
4.4.10.10. Counting triangle	1783
4.4.10.11. Tree depth	1785

4.4.11. Tools	1787
4.4.11.1. SQL script	1787
4.4.12. Financials	1788
4.4.12.1. Binning	1788
4.4.12.2. Data conversion	1790
4.4.12.3. Scorecard training	1791
4.4.12.4. Scorecard prediction	1798
4.4.12.5. PSI	1800
4.4.13. Video intelligence platform (must be separately activa...)	1802
4.4.13.1. Video preprocessing	1802
4.4.13.1.1. Convert datasets to TFRecord files	1802
4.4.13.1.2. General video preprocessing	1813
4.4.13.2. Offline training	1814
4.4.13.2.1. Train a model for semantic image segmentation	1814
4.4.13.2.2. Train an image detection model	1820
4.4.13.2.3. Train an image classification model	1829
4.4.13.2.4. Train a video classification model	1838
4.4.13.3. Offline prediction	1842
4.4.13.3.1. General image prediction	1842
4.4.13.3.2. General video prediction	1863
4.4.13.3.3. ASR prediction	1876
4.5. OpenAPI	1878
4.5.1. Query PMML models	1878
4.5.2. Query detailed information about a PMML model	1881
4.5.3. Download PMML models	1883
4.5.3.1. Generate a download URL for a model	1883
4.5.3.2. Query a model URL generation task	1884
4.5.4. SDKs	1886

4.6. Data labeling	1888
4.6.1. Register a dataset	1888
4.6.2. Data labeling templates	1890
4.6.3. Create a labeling job	1896
4.6.4. Label images	1897
4.7. EAS user guide	1898
4.7.1. EAS overview	1898
4.7.2. Client	1898
4.7.3. User authentication	1898
4.7.4. Upload files	1899
4.7.5. Create a service	1899
4.7.6. Local debugging	1902
4.7.7. Modify configurations	1903
4.7.8. Modify a service	1903
4.7.9. Delete a service	1903
4.7.10. Switch service version	1904
4.7.11. View service list	1904
4.7.12. View service information	1904
4.7.13. View service processes	1905
4.7.14. Call a prediction service	1906
4.7.15. Use Java or C++ to develop a model service	1906
4.7.15.1. What are processors	1906
4.7.15.2. C/C++ processor	1907
4.7.15.3. Java processor	1909
4.8. AutoML (must be activated separately)	1909
4.8.1. Automatic parameter tuning with AutoML	1909
4.8.2. Parameter tuning methods	1911
4.9. Terms and acronyms	1912

4.9.1. Terms	1912
4.9.2. Acronyms	1913
5.E-MapReduce (EMR)	1914
5.1. What is E-MapReduce?	1914
5.2. Introduction	1914
5.2.1. Instructions	1914
5.2.2. Introduction	1914
5.2.2.1. Software configuration	1914
5.2.2.2. Software requirements	1914
5.2.2.3. Supported components	1914
5.2.2.4. Introduction to components	1916
5.3. Log on to the EMR console	1918
5.4. Cluster management	1919
5.4.1. Cluster planning	1919
5.4.1.1. Gateway clusters	1919
5.4.2. Cluster configuration	1919
5.4.2.1. Create a cluster	1919
5.4.2.2. Create a gateway cluster	1923
5.4.2.3. Cluster list and details	1925
5.4.2.4. View the running status of services	1927
5.4.2.5. Configure component parameters	1928
5.4.3. Restart a service or component	1929
5.4.4. View information about instances	1930
6.DataHub	1931
6.1. What is DataHub?	1931
6.2. Usage notes	1931
6.3. Quick Start	1932
6.3.1. Overview	1932

6.3.2. Log on to the DataHub console	1933
6.3.3. Create a project	1934
6.3.4. Create a topic	1934
6.3.5. Sample data	1935
6.3.6. Create a DataConnector	1935
6.3.7. Create a subscription	1936
6.4. Access Control	1936
6.4.1. Overview	1936
6.4.2. DataHub resources in RAM	1936
6.4.3. API	1937
6.4.4. Conditions	1938
6.4.5. Sample RAM authorization policy content	1939
6.4.5.1. AliyunDataHubFullAccess	1939
6.4.5.2. AliyunDataHubReadOnlyAccess	1939
6.5. Data Acquisition	1939
6.5.1. Overview	1939
6.5.2. Fluentd	1940
6.5.3. Logstash	1943
6.5.4. Oracle GoldenGate	1948
6.6. Data synchronization	1956
6.6.1. Overview	1956
6.6.2. Synchronize data to MaxCompute	1956
6.6.2.1. Create a DataConnector	1956
6.6.2.2. View data synchronization details	1957
6.7. Metric statistics	1958
6.8. Data subscription	1958
6.8.1. Overview	1958
6.8.2. Create a subscription	1958

6.8.3. Use case	1959
6.9. Collaborative consumption	1962
6.9.1. Note	1962
6.9.2. Overview	1962
6.9.3. Maven dependencies and JDK	1963
6.9.4. Use case	1963
6.9.5. Usage notes	1966
7.Quick BI	1967
7.1. What is Quick BI?	1967
7.2. Log on to the Quick BI console	1967
7.3. Access mode	1968
7.4. Data modeling	1968
7.4.1. Data modeling	1969
7.4.2. Data sources	1969
7.4.2.1. Overview	1969
7.4.2.2. Cloud data sources	1970
7.4.2.2.1. Add the IP addresses of a Quick BI cluster to a ...	1970
7.4.2.2.2. Add a MaxCompute data source	1971
7.4.2.2.3. Add an ApsaraDB RDS for MySQL data source	1972
7.4.2.2.4. Add an ApsaraDB RDS for SQL Server data sour...	1974
7.4.2.2.5. Add an AnalyticDB for MySQL 2.0 data source	1976
7.4.2.2.6. Add a HybridDB for MySQL data source	1977
7.4.2.2.7. Add an AnalyticDB for PostgreSQL data source	1978
7.4.2.2.8. Add an AnalyticDB for PostgreSQL data source	1979
7.4.2.2.9. Add an ApsaraDB RDS for PPAS data source	1980
7.4.2.2.10. Add a cloud data source Data Lake Analytics	1982
7.4.2.2.11. Add a Hive data source	1982
7.4.2.2.12. Add a PolarDB-X data source	1983

7.4.2.2.13. Add a cloud Presto data source	1984
7.4.2.2.14. Add an AnalyticDB for MySQL V3.0 data source	1985
7.4.2.2.15. Add a PolarDB for MySQL data source	1986
7.4.2.3. User-created data sources	1988
7.4.2.3.1. Add a user-created MySQL data source	1988
7.4.2.3.2. Add a user-created SQL Server data source	1990
7.4.2.3.3. Add a user-created PostgreSQL data source	1992
7.4.2.3.4. Add a user-created Oracle data source	1994
7.4.2.3.5. Add a self-managed Hive data source	1996
7.4.2.3.6. Add a self-managed Vertica data source	1997
7.4.2.3.7. Add a user-created IBM DB2 LUW data source	1998
7.4.2.3.8. Add a user-created SAP IQ (Sybase IQ) data sou...	1999
7.4.2.3.9. Add a user-created SAP HANA data source	2001
7.4.2.3.10. Add a user-created Presto data source	2002
7.4.2.4. Go to the Data Sources page	2004
7.4.2.5. Create a data source	2004
7.4.2.6. Edit a data source	2005
7.4.2.7. Replace a data source	2006
7.4.2.8. Delete a data source	2006
7.4.2.9. Search for a data source	2007
7.4.2.10. Search for a table under a data source	2008
7.4.2.11. View the details of a table under a data source	2008
7.4.3. Datasets	2009
7.4.3.1. Overview of datasets	2009
7.4.3.2. Create datasets	2009
7.4.3.2.1. Create a dataset based on a data source	2009
7.4.3.2.2. Create a dataset by uploading a file	2010
7.4.3.2.3. Create a dataset by using an SQL statement for...	2010

7.4.3.3. Specify a method to name dimensions and measure...	2013
7.4.3.4. Edit a dataset	2014
7.4.3.4.1. Edit a dimension	2014
7.4.3.4.2. Edit a measure	2016
7.4.3.4.3. Toolbar and shortcut menu	2017
7.4.3.4.4. Preview data	2018
7.4.3.4.5. Table join and examples	2018
7.4.3.4.6. Calculated fields	2020
7.4.3.4.6.1. Overview	2020
7.4.3.4.6.2. Rules for using calculated fields	2021
7.4.3.4.6.3. Types of calculated measures	2021
7.4.3.4.6.4. Expressions of calculated fields	2021
7.4.3.4.6.5. Add a calculated field	2022
7.4.3.4.7. Add a grouping field	2025
7.4.3.5. Rename a dataset	2026
7.4.3.6. Search for a dataset	2026
7.4.3.7. Transfer a dataset	2027
7.4.3.8. Copy a dataset from one workspace to another	2028
7.4.3.9. Configure the dataset cache	2029
7.4.3.10. Create a dataset folder	2030
7.4.3.11. Rename a dataset folder	2030
7.4.3.12. Delete a dataset	2031
7.4.3.13. Configure row-level permissions	2032
7.5. Dashboards	2032
7.5.1. Dashboard overview	2032
7.5.1.1. Dashboard features	2032
7.5.1.2. Chart types and scenarios	2032
7.5.1.3. Data elements of a chart	2033

7.5.2. Access a dashboard	2037
7.5.3. Areas of a dashboard	2037
7.5.3.1. Overview	2037
7.5.3.2. Dataset selection area	2038
7.5.3.2.1. Switch datasets	2038
7.5.3.2.2. Search for a dimension or measure	2039
7.5.3.3. Dashboard graphic design area	2039
7.5.3.3.1. Select fields	2039
7.5.3.3.2. Use the color legend	2040
7.5.3.3.3. Sort field data	2041
7.5.3.3.4. Configure advanced calculations	2043
7.5.3.3.5. Configure the comparison feature	2044
7.5.3.3.6. Filter field data	2052
7.5.3.3.7. Perform drill-down analysis	2053
7.5.3.3.8. Filter interaction	2056
7.5.3.3.9. Metric analysis	2058
7.5.3.4. Dashboard display area	2062
7.5.3.4.1. Overview	2063
7.5.3.4.2. Toolbar	2063
7.5.3.4.3. Adjust chart positions	2063
7.5.3.4.4. View chart data	2063
7.5.3.4.5. Change chart types	2064
7.5.3.4.6. Add to Favorites	2065
7.5.3.4.7. Delete a chart	2065
7.5.3.4.8. Widgets	2066
7.5.3.4.8.1. Overview	2066
7.5.3.4.8.2. Filter bar	2066
7.5.3.4.8.3. Compound Query Control	2076

7.5.3.4.8.4. Text Area	2080
7.5.3.4.8.5. IFrame	2080
7.5.3.4.8.6. Tab	2081
7.5.3.4.8.7. Image	2083
7.5.4. Create a chart on the dashboard	2083
7.5.4.1. Create a line chart	2083
7.5.4.2. Create an area chart	2085
7.5.4.3. Create a vertical bar chart	2086
7.5.4.4. Create a waterfall chart	2089
7.5.4.5. Create a horizontal bar chart	2091
7.5.4.6. Create a progress bar	2093
7.5.4.7. Create a combination chart	2094
7.5.4.8. Create a pie chart	2095
7.5.4.9. Create a bubble map	2097
7.5.4.10. Create a colored map	2099
7.5.4.11. Create a cross table	2100
7.5.4.12. Create a pivot table	2104
7.5.4.13. Create a gauge	2105
7.5.4.14. Create a radar chart	2107
7.5.4.15. Create a facet scatter chart	2108
7.5.4.16. Create a scatter chart	2109
7.5.4.17. Create a bubble chart	2111
7.5.4.18. Create a funnel chart	2112
7.5.4.19. Create a kanban	2114
7.5.4.20. Create an indicator trend chart	2115
7.5.4.21. Create a treemap	2118
7.5.4.22. Create a polar diagram	2120
7.5.4.23. Create a word cloud	2122

7.5.4.24. Create a butterfly chart	2123
7.5.4.25. Create a hierarchy chart	2125
7.5.4.26. Create a flow analysis chart	2133
7.5.4.27. Create a sankit diagram	2134
7.5.4.28. Create a Leaderboard	2135
7.5.4.29. Create a ticker board	2137
7.5.5. Theme design	2139
7.5.6. Full screen mode	2140
7.5.7. Search for a dashboard	2145
7.5.8. Create a dashboard folder	2145
7.5.9. Rename a dashboard folder	2145
7.5.10. View a dashboard	2146
7.5.11. Share a dashboard	2147
7.5.12. Make a dashboard public	2148
7.5.13. Publish a dashboard	2149
7.6. Workbooks	2150
7.6.1. Overview	2150
7.6.2. Create a workbook	2151
7.6.3. Switch to another dataset	2152
7.6.4. Search for a dimension or measure	2153
7.6.5. Configure a workbook	2154
7.6.6. Manage a workbook	2161
7.6.7. Publish a workbook	2163
7.6.8. Create a workbook folder	2164
7.6.9. Rename a workbook folder	2165
7.6.10. Delete a workbook folder	2165
7.7. BI portals	2166
7.7.1. Overview	2166

7.7.2. Create a BI portal	2166
7.7.3. Configure a portal	2166
7.7.4. Configure menus	2168
7.7.5. Manage BI portals	2170
7.7.6. Manage menu permissions of a BI portal	2171
7.8. Organization	2173
7.8.1. Overview	2173
7.8.2. Create an organization	2173
7.8.3. Modify organization information	2173
7.8.4. Leave an organization	2174
7.8.5. Add a member	2175
7.8.6. Manage member tags	2181
7.8.7. Edit a member	2182
7.8.8. Remove a member	2183
7.8.9. Query the workspace to which a user belongs	2184
7.8.10. Search for a member of an organization	2184
7.8.11. Create a user group	2184
7.8.12. Manage theme templates	2185
7.8.13. Workspaces	2186
7.8.13.1. Overview	2186
7.8.13.2. What is a workspace?	2186
7.8.13.3. Differences between a personal workspace and a g...	2189
7.8.14. Create a workspace	2189
7.8.15. Modify information about a workspace	2190
7.8.16. Leave a workspace	2191
7.8.17. Transfer a workspace to another member	2192
7.8.18. Delete a workspace	2192
7.8.19. Add a member to a workspace	2192

7.8.20. Edit settings of a workspace member	2193
7.8.21. Search for a member in a workspace	2194
7.8.22. Delete a member from a workspace	2194
7.9. Permissions	2194
7.9.1. Overview of permissions	2194
7.9.2. Manage data objects	2195
7.9.3. Manage row-level permissions	2195
7.9.4. Configure menu permissions for a BI portal	2198
7.9.5. Share a data object in a personal workspace	2199
7.9.6. Share a data object in a workspace	2200
7.9.7. Publish data objects that are stored in a personal work...	2200
7.9.8. Make a data object in a workspace public	2201
7.10. Report statistics	2201
7.10.1. Usage statistics	2201
7.10.2. Access statistics	2202
7.10.3. Lineage analysis	2202
7.11. Open service	2202
7.11.1. Overview of the open service	2203
7.11.2. Report embedment	2203
8. Graph Analytics	2204
8.1. What is Graph Analytics?	2204
8.2. Quick Start	2204
8.2.1. Log on to Administration Console of Graph Analytics	2204
8.2.2. Create data sources	2205
8.2.3. Create OLEP models for tables	2208
8.2.4. Add OLEP table columns	2217
8.2.5. Configure object properties and business parameters	2219
8.2.6. Configure link properties and business parameters	2222

8.2.7. Configure event properties and business parameters	2226
8.2.8. Log on to Analytics Workbench	2229
8.2.9. Create an analysis	2230
8.2.10. View analyses	2231
8.3. Source tables	2232
8.3.1. Data sources	2232
8.3.1.1. Create data sources	2232
8.3.1.2. View data sources	2234
8.3.1.3. Modify a data source	2235
8.3.1.4. Delete a data source	2235
8.3.2. OLEP tables	2235
8.3.2.1. Create OLEP models for tables	2235
8.3.2.2. View an OLEP table	2245
8.3.2.3. Edit OLEP tables	2246
8.3.2.4. Remove an OLEP table	2247
8.3.3. OLEP table columns	2248
8.3.3.1. Add OLEP table columns	2248
8.3.3.2. Edit OLEP table columns	2249
8.3.3.3. Remove OLEP table columns	2250
8.4. Dictionaries	2251
8.4.1. Create a dictionary	2252
8.4.2. Modify a dictionary	2254
8.4.3. Delete a dictionary	2254
8.5. Object information	2255
8.5.1. Object groups	2255
8.5.1.1. Create an object group	2255
8.5.1.2. View object groups and objects	2256
8.5.1.3. Modify object groups and objects	2257

8.5.1.4. Delete object groups and objects	2259
8.5.2. Objects	2260
8.5.2.1. Create an object	2260
8.5.2.2. Configure object properties and business parameters	2262
8.5.2.3. Enable and disable an object	2265
8.5.2.4. Modify an object	2266
8.5.2.5. Delete an object	2267
8.6. Link information	2268
8.6.1. Link groups and links	2268
8.6.1.1. Create a link group	2268
8.6.1.2. View links and link groups	2269
8.6.1.3. Modify a link or link group	2270
8.6.1.4. Delete a link or link group	2272
8.6.2. First-degree links	2273
8.6.2.1. Create a first-degree link	2273
8.6.2.2. Configure link properties and business parameters	2275
8.6.3. Create a second-degree link	2279
8.6.4. Create a multi-degree link	2283
8.7. Event information	2285
8.7.1. Event groups	2285
8.7.1.1. Create an event group	2285
8.7.1.2. View an event group	2286
8.7.1.3. Modify an event group	2286
8.7.1.4. Delete an event group	2287
8.7.2. Events	2287
8.7.2.1. Create an event	2287
8.7.2.2. Configure event properties and business parameters	2288
8.7.2.3. Enable and disable an event	2291

8.7.2.4. View an event	2292
8.7.2.5. Modify an event	2293
8.7.2.6. Delete an event	2293
8.8. View the business graph	2294
8.9. Advanced configurations	2294
8.9.1. Manage a system model	2294
8.9.2. Configure a search item	2297
8.9.3. System settings	2299
8.9.3.1. Configure components	2299
8.9.3.2. Technical parameters	2301
8.9.3.2.1. Path analysis settings	2301
8.9.3.2.2. Quick extension settings	2302
8.9.3.2.3. Maximum node settings	2302
8.9.3.3. Business parameters	2303
8.9.3.3.1. Add double-click link settings	2303
8.9.3.3.2. Double-click-disabled object settings	2304
8.9.3.3.3. Object grouping settings	2305
8.9.3.3.4. Configure lineage analysis	2306
8.9.3.3.5. Intimacy measurement settings	2306
8.9.3.3.6. Redirect URL settings	2307
8.9.3.4. Object icons	2307
8.9.3.4.1. Upload an object icon	2307
8.9.3.4.2. Modify an object icon	2308
8.9.3.4.3. Delete an object icon	2308
8.9.4. System labels	2308
8.9.4.1. Create a group	2308
8.9.4.2. Create a system label	2309
8.9.4.3. Modify a system label	2310

8.9.4.4. Delete a system label	2311
8.9.5. System operations and maintenance	2311
8.9.5.1. Audit logs	2311
8.9.6. View server clusters	2312
8.10. Import data	2312
8.10.1. Model list	2312
8.10.1.1. Model overview	2312
8.10.1.2. View models	2313
8.10.1.3. Create models	2313
8.10.1.4. Modify model names	2315
8.10.1.5. Download a model	2315
8.10.1.6. Delete a model	2316
8.10.2. Import data	2316
8.10.3. Data list	2318
8.10.3.1. View data	2318
8.10.3.2. Edit a data name	2318
8.10.3.3. Import data to Graph	2318
8.10.3.4. Delete data	2319
8.11. Search	2319
8.11.1. Search	2319
8.11.2. Simple search	2320
8.11.3. Advanced search	2321
8.11.4. View and analyze search results	2322
8.12. Graph	2324
8.12.1. Graph	2324
8.12.2. Analysis types	2324
8.12.3. Create an analysis	2325
8.12.4. Add a node	2326

8.12.5. Delete nodes, links, and events	2328
8.12.6. Link extension	2328
8.12.7. Graphic operations	2330
8.12.7.1. Move canvases	2330
8.12.7.2. Zoom in and zoom out canvases	2331
8.12.7.3. Undo and redo operations	2331
8.12.7.4. View thumbnails	2332
8.12.7.5. Right-click menu	2333
8.12.8. Analyze	2336
8.12.8.1. Group Analysis	2336
8.12.8.2. Common neighbor analysis	2338
8.12.8.3. Lineage analysis	2339
8.12.8.4. Path analysis	2341
8.12.8.5. Backbone analysis	2343
8.12.8.6. Intimacy measurements	2344
8.12.9. Lock or unlock nodes	2345
8.12.10. Network analysis	2346
8.12.11. Closed-loop mining	2349
8.12.12. Layouts	2350
8.12.13. Flag and unflag nodes	2354
8.12.14. Labels	2354
8.12.14.1. Label types	2354
8.12.14.2. User labels	2355
8.12.14.3. Add user labels	2356
8.12.14.4. View labels	2357
8.12.14.5. Click likes and delete likes	2357
8.12.14.6. Edit user labels	2358
8.12.14.7. Delete user labels	2359

8.12.15. Save analysis	2360
8.12.16. Print graph areas	2360
8.12.17. Share analyses	2361
8.12.18. Behavior chronology	2362
8.12.18.1. Details	2362
8.12.18.2. Behavior analysis	2363
8.12.18.3. Chronology analysis	2364
8.12.19. Property statistics	2366
8.12.19.1. Details	2366
8.12.19.2. Statistics	2367
8.12.19.3. Property information	2370
8.12.19.4. Secondary filtering	2372
8.13. File Center	2373
8.13.1. View and manage all analyses	2373
8.13.2. View and manage your files	2374
8.13.3. My shared items	2376
8.13.3.1. Overview	2376
8.13.3.2. View and manage shared files	2376
8.13.3.3. Edit a shared file	2378
8.13.3.4. Publish a version	2379
8.13.3.5. Automatically merge files	2381
8.13.4. View and manage received shared files	2382
8.14. Intelligent Network	2383
8.14.1. Intelligent Network overview	2383
8.14.2. Patterns	2384
8.14.2.1. Create patterns	2384
8.14.2.2. View patterns	2389
8.14.2.3. Modify patterns	2390

8.14.2.4. Set private patterns to public patterns	2392
8.14.2.5. Delete a pattern	2393
8.14.3. Tasks	2394
8.14.3.1. Create a task	2394
8.14.3.2. Check the task	2395
8.14.3.3. Modify a task	2397
8.14.3.4. Execute the task and view the result	2399
8.14.3.5. Set a private task as a public task	2401
8.14.3.6. Delete a task	2402
8.15. Examples	2403
8.15.1. Tax industry case studies	2403
8.16. FAQ	2408
9. Dataphin	2410
9.1. What is Dataphin?	2410
9.2. Usage notes	2410
9.3. Quick start	2411
9.3.1. Instructions for system administrators	2411
9.3.2. Log on to the Dataphin console	2412
9.3.3. Preparations	2413
9.3.4. Ingest data	2413
9.3.5. Integrate data	2414
9.3.6. Define data standardization objects	2418
9.3.7. Create data modeling objects	2419
9.3.8. Generate retroactive data	2428
9.3.9. Verify data	2430
9.3.10. Publish data	2431
9.3.11. Verify the scheduling result	2432
9.4. Management Center	2433

9.4.1. Initialize metadata	2433
9.4.2. Manage members	2434
9.4.3. Configure the computing engine settings	2436
9.4.4. Configure the intelligent data engine	2437
9.4.4.1. Configure data skew optimization	2437
9.4.4.2. Configure code optimization	2438
9.4.4.3. Set custom task parameters	2439
9.5. Notifications	2440
9.5.1. View and handle messages	2440
9.5.2. View and handle tasks	2441
9.6. Alert Center	2442
9.6.1. Overview	2442
9.6.2. Terms	2443
9.6.3. Events	2443
9.6.4. Push records	2447
9.6.5. Shift schedules	2449
9.6.5.1. Overview	2449
9.6.5.2. Create a shift schedule	2450
9.6.5.3. View details of a shift schedule	2452
9.6.5.4. Modify a shift schedule	2452
9.6.5.5. Delete a shift schedule	2453
9.7. Data warehouse planning	2453
9.7.1. Overview	2453
9.7.2. Business unit management	2454
9.7.2.1. Create business units	2454
9.7.2.2. Modify a business unit	2457
9.7.2.3. Modify business unit parameters	2457
9.7.2.4. Data domain management	2459

9.7.2.4.1. Create a data domain	2459
9.7.2.4.2. Modify a data domain	2460
9.7.2.4.3. Delete a data domain	2460
9.7.2.5. Delete a business unit	2461
9.7.3. Statistical period management	2461
9.7.3.1. Create a statistical period	2461
9.7.3.2. Modify a statistical period	2463
9.7.3.3. Delete a statistical period	2463
9.7.4. Project management	2464
9.7.4.1. Overview	2464
9.7.4.2. Create projects	2465
9.7.4.3. Configure projects	2470
9.7.4.4. Delete projects	2472
9.7.5. Maintenance and upgrade	2473
9.7.5.1. Important notes for upgrade	2473
9.7.5.2. Maintain and upgrade business units	2474
9.7.5.3. View maintenance and upgrade logs	2475
9.7.6. Data source	2476
9.7.6.1. Overview	2476
9.7.6.2. Create batch processing data sources	2477
9.7.6.2.1. Create a MaxCompute data source	2477
9.7.6.2.2. Create a MySQL data source	2478
9.7.6.2.3. Create an SQL Server data source	2479
9.7.6.2.4. Create a PostgreSQL data source	2480
9.7.6.2.5. Create an Oracle data source	2481
9.7.6.2.6. Create an HDFS data source	2482
9.7.6.2.7. Create an FTP data source	2483
9.7.6.2.8. Create a Hive data source	2484

9.7.6.2.9. Create an Elasticsearch data source	2486
9.7.6.2.10. Create a MongoDB data source	2487
9.7.6.2.11. Create an AnalyticDB data source	2488
9.7.6.2.12. Create a PolarDB-X data source	2489
9.7.6.2.13. Create a Vertica data source	2490
9.7.6.2.14. Create an HBase data source	2491
9.7.6.2.15. Create an AnalyticDB for MySQL V3.0 data sou...	2492
9.7.6.2.16. Create an AnalyticDB for PostgreSQL data sour...	2493
9.7.6.2.17. Create a LogHub data source	2494
9.7.6.3. Create stream processing data sources	2495
9.7.6.3.1. Create a DataHub data source	2495
9.7.6.3.2. Create a Log Service data source	2496
9.7.6.3.3. Create an ApsaraDB for HBase data source	2497
9.7.6.3.4. Create a Kafka data source	2498
9.7.6.3.5. Create a Tablestore data source	2499
9.7.6.3.6. Create a RocketMQ data source	2500
9.7.6.4. Modify a data source	2501
9.7.6.5. Test a data source	2502
9.7.6.6. Change the owner of a data source	2503
9.7.6.7. Delete a data source	2503
9.7.7. Computing engines	2504
9.7.7.1. Overview	2504
9.7.7.2. Create a batch processing computing engine	2504
9.7.7.3. Create a stream processing computing engine	2506
9.7.7.4. Test a computing engine	2507
9.7.7.5. Modify a computing engine	2507
9.7.7.6. Change the owner of a computing engine	2508
9.7.7.7. Delete a computing engine	2508

9.8. Data ingestion	2509
9.8.1. Overview	2509
9.8.2. Data integration	2509
9.8.2.1. Overview	2509
9.8.2.2. Manage folders	2510
9.8.2.3. Upload a JSON file for creating an offline migratio...	2511
9.8.2.4. Manage offline migration pipelines	2512
9.8.2.4.1. Create and configure an offline migration pipeli...	2512
9.8.2.4.2. Edit an offline migration pipeline	2521
9.8.2.4.3. Rename an offline migration pipeline	2522
9.8.2.4.4. Move an offline migration pipeline	2523
9.8.2.4.5. View historical version information about an off...	2523
9.8.2.4.6. Unpublish and delete an offline migration pipel...	2524
9.8.2.5. Manage offline database migration tasks	2525
9.8.2.5.1. Create and configure an offline database migrat...	2525
9.8.2.5.2. Manage offline migration pipelines	2527
9.8.2.5.3. Rename an offline database migration task	2530
9.8.2.5.4. Delete an offline database migration task	2531
9.8.2.6. Input components	2531
9.8.2.6.1. Manage MySQL input components	2531
9.8.2.6.2. Manage Maxcompute input components	2534
9.8.2.6.3. Manage Vertica input components	2537
9.8.2.6.4. Manage DRDS input components	2539
9.8.2.6.5. Manage PostgreSQL input components	2542
9.8.2.6.6. Manage SQL Server input components	2545
9.8.2.6.7. Manage ORACLE input components	2548
9.8.2.6.8. Manage FTP input components	2551
9.8.2.6.9. Manage HDFS input components	2554

9.8.2.6.10. Manage Hive input components	2557
9.8.2.6.11. Manage Hbase input components	2560
9.8.2.6.12. Manage MongoDB input components	2563
9.8.2.6.13. Manage LogicalTable input components	2566
9.8.2.6.14. Manage ADB for MySQL3.0 input components	2568
9.8.2.6.15. Manage ADB for PostgreSQL input components	2571
9.8.2.6.16. Manage LogHub input components	2574
9.8.2.7. Conversion components	2578
9.8.2.7.1. Manage Field Selection components	2578
9.8.2.7.2. Manage Field Computing components	2580
9.8.2.7.3. Manage Filtering components	2582
9.8.2.8. Process components	2585
9.8.2.8.1. Manage Access Limit components	2585
9.8.2.8.2. Manage Conditional Distribution components	2587
9.8.2.9. Output components	2589
9.8.2.9.1. Manage MySQL output components	2589
9.8.2.9.2. Manage DRDS output components	2591
9.8.2.9.3. Manage Maxcompute output components	2594
9.8.2.9.4. Manage SQL Server output components	2597
9.8.2.9.5. Manage ORACLE output components	2600
9.8.2.9.6. Manage PostgreSQL output components	2602
9.8.2.9.7. Manage Vertica output components	2605
9.8.2.9.8. Manage FTP output components	2607
9.8.2.9.9. Manage HDFS output components	2610
9.8.2.9.10. Manage MongoDB output components	2613
9.8.2.9.11. Manage Hbase output components	2616
9.8.2.9.12. Manage Hive output components	2619
9.8.2.9.13. Manage ElasticSearch output components	2622

9.8.2.9.14. Manage ADB for MySQL2.0 output components	2625
9.8.2.9.15. Manage ADB for MySQL3.0 output components	2628
9.8.2.9.16. Manage ADB for PostgreSQL output componen...	2631
9.8.3. Data synchronization	2633
9.8.3.1. Create a folder for storing sync tasks	2634
9.8.3.2. Create a destination table for data synchronization	2635
9.8.3.3. Create a sync task	2638
9.8.3.4. Configure a sync task	2640
9.8.3.5. Configure a scheduling policy	2643
9.8.3.6. Run a one-time sync task	2647
9.8.3.7. Verify a sync task	2648
9.9. Data development	2649
9.9.1. Standards and modeling	2649
9.9.1.1. Overview	2649
9.9.1.2. Data standardization: Dimensions	2651
9.9.1.2.1. Create a dimension	2651
9.9.1.2.2. View the logical dimension table of a dimension	2656
9.9.1.2.3. Modify a dimension	2656
9.9.1.2.4. Unpublish and delete a dimension	2657
9.9.1.3. Data standardization: Business processes	2659
9.9.1.3.1. Create a business process	2659
9.9.1.3.2. Modify a business process	2660
9.9.1.3.3. Clone a business process	2661
9.9.1.3.4. Create a logical table	2662
9.9.1.3.5. View logical fact tables related to a business pr...	2663
9.9.1.3.6. Delete a business process	2663
9.9.1.4. Logical tables: Logical dimension tables	2664
9.9.1.4.1. Edit model information	2664

9.9.1.4.2. Edit a central table	2666
9.9.1.4.3. Configure a logical table conversion task	2670
9.9.1.4.4. Configure a scheduling policy	2671
9.9.1.4.5. Unpublish and delete a logical dimension table	2674
9.9.1.5. Logical tables: Logical fact tables	2675
9.9.1.5.1. Create a logical fact table	2675
9.9.1.5.2. Configure a logical table conversion task	2680
9.9.1.5.3. Modify a logical fact table	2681
9.9.1.5.4. Configure a scheduling policy	2682
9.9.1.5.5. View historical version information	2685
9.9.1.5.6. Unpublish and delete a logical fact table	2686
9.9.1.6. Data standardization: Atomic metrics	2687
9.9.1.6.1. Create an atomic metric	2687
9.9.1.6.2. Modify an atomic metric	2691
9.9.1.6.3. View atomic metrics that share the same source... ..	2692
9.9.1.6.4. Clone an atomic metric	2693
9.9.1.6.5. View derived metrics created based on an atomi... ..	2694
9.9.1.6.6. Create a derived metric	2694
9.9.1.6.7. Unpublish and delete an atomic metric	2695
9.9.1.7. Data standardization: Business filters	2697
9.9.1.7.1. Create a business filter	2697
9.9.1.7.2. Clone a business filter	2698
9.9.1.7.3. View business filters that share the same source	2699
9.9.1.7.4. View derived metrics created based on a busines... ..	2700
9.9.1.7.5. Modify a business filter	2700
9.9.1.7.6. Unpublish and delete a business filter	2701
9.9.1.8. Data standardization: Derived metrics	2703
9.9.1.8.1. Create a derived metric	2703

9.9.1.8.2. Modify a derived metric	2705
9.9.1.8.3. Unpublish and delete a derived metric	2705
9.9.1.8.4. View the logical aggregate table associated with.....	2707
9.9.1.9. Logical tables: Logical aggregate tables	2708
9.9.1.9.1. Create a logical aggregate table	2708
9.9.1.9.2. Modify a logical aggregate table	2709
9.9.1.9.3. Configure a logical table conversion task	2712
9.9.1.9.4. View details of a logical aggregate table	2713
9.9.1.9.5. Delete a logical aggregate table	2714
9.9.1.10. Standards and modeling: Modeling engine	2715
9.9.2. Batch processing	2716
9.9.2.1. Overview	2716
9.9.2.2. Manage resources	2717
9.9.2.2.1. Create a folder	2717
9.9.2.2.2. Create a resource	2718
9.9.2.2.3. Modify a resource	2719
9.9.2.2.4. Move a resource	2720
9.9.2.2.5. Delete a resource	2720
9.9.2.3. Manage batch processing functions	2721
9.9.2.3.1. Create a folder	2721
9.9.2.3.2. Create a UDF	2722
9.9.2.3.3. Modify a UDF	2724
9.9.2.3.4. Move a UDF	2725
9.9.2.3.5. Delete a UDF	2725
9.9.2.4. Create a folder for storing batch processing tasks	2726
9.9.2.5. Create batch processing tasks	2726
9.9.2.5.1. Create a batch processing task of the MAX_COM.....	2727
9.9.2.5.2. Create a batch processing task of the MAX_COM.....	2728

9.9.2.5.3. Create a batch processing task of the SPARK_JA...	2729
9.9.2.5.4. Create a batch processing task of the SHELL ty...	2730
9.9.2.5.5. Create a batch processing task of the PYTHON ...	2732
9.9.2.5.6. Create a zero load node of the VIRTUAL type	2733
9.9.2.6. Configure a scheduling policy	2735
9.9.2.7. Modify a batch processing task	2739
9.9.2.8. Unpublish and delete a batch processing task	2740
9.9.2.9. View the version information about a batch process...	2741
9.9.2.10. Move a batch processing task	2742
9.9.2.11. Rename a batch processing task	2742
9.9.3. Stream processing	2742
9.9.3.1. Overview	2743
9.9.3.2. Manage resources	2743
9.9.3.2.1. Create a folder	2743
9.9.3.2.2. Create a resource	2744
9.9.3.2.3. Modify a resource	2746
9.9.3.2.4. Move a resource	2746
9.9.3.2.5. Delete a resource	2747
9.9.3.3. Manage stream processing functions	2747
9.9.3.3.1. Create a folder for storing stream processing fu...	2747
9.9.3.3.2. Create a UDF	2748
9.9.3.3.3. Modify a UDF	2750
9.9.3.3.4. Move a UDF	2750
9.9.3.3.5. Delete a UDF	2751
9.9.3.4. Manage stream processing metatables	2752
9.9.3.4.1. Create a folder for storing stream metatables	2752
9.9.3.4.2. Create a stream metatable	2753
9.9.3.4.3. Modify a stream metatable	2755

9.9.3.4.4. View the version information about a stream m...	2756
9.9.3.4.5. Move a stream metatable	2757
9.9.3.4.6. Delete a stream metatable	2757
9.9.3.5. Manage stream processing templates	2758
9.9.3.5.1. Create a folder for storing stream processing te...	2758
9.9.3.5.2. Create a stream processing template	2759
9.9.3.5.3. Modify a stream processing template	2760
9.9.3.5.4. View the version information about a stream pr...	2761
9.9.3.5.5. Move a stream processing template	2762
9.9.3.5.6. Delete a stream processing template	2762
9.9.3.6. Manage stream processing tasks	2763
9.9.3.6.1. Create a folder for storing stream processing tas...	2763
9.9.3.6.2. Create a stream processing task of the FLINK_S...	2764
9.9.3.6.3. Create a stream processing task of the FLINK_T...	2766
9.9.3.6.4. Modify a stream processing task	2769
9.9.3.6.5. View the version information about a stream pr...	2770
9.9.3.6.6. Move a stream processing task	2770
9.9.3.6.7. Unpublish or delete a stream processing task	2771
9.9.4. Ad hoc query	2772
9.9.4.1. Overview	2772
9.9.4.2. Create an ad hoc query task	2772
9.9.4.3. Manage ad hoc query tasks	2773
9.10. Data distilling	2775
9.10.1. Overview	2775
9.10.2. Behavior Engine	2776
9.10.2.1. Overview	2776
9.10.2.2. Manage behavioral domains	2779
9.10.2.3. Manage lines-of-business	2781

9.10.2.4. Manage actions	2783
9.10.2.5. Manage objects	2785
9.10.2.6. Manage object attributes	2786
9.10.2.7. Create a behavior rule	2788
9.10.2.8. Manage behavior rules	2795
9.10.2.9. View behavioral data	2800
9.10.3. Tag Engine	2800
9.10.3.1. Create a factory tag	2800
9.10.3.2. Manage factory tags	2805
9.10.3.3. Manage the logical tag table	2808
9.10.3.4. Manage tag categories	2812
9.11. Task publishing	2814
9.11.1. Publishing management	2814
9.12. Scheduling center	2816
9.12.1. Overview	2816
9.12.2. Global management	2818
9.12.2.1. One-time tasks	2818
9.12.2.2. Recurring tasks	2821
9.12.2.3. Stream processing tasks	2823
9.12.2.4. One-time instances	2826
9.12.2.5. Recurring instances	2827
9.12.2.6. Stream processing instances	2831
9.12.2.7. Retroactive data generation instances	2833
9.12.3. Logical tables	2836
9.12.3.1. Logical table tasks	2836
9.12.3.2. Logical table task instances	2837
9.12.4. Distilling management	2839
9.12.4.1. Behavior rule tasks	2839

9.12.4.2. Tag tasks	2842
9.12.4.3. Behavior rule task instances	2845
9.12.4.4. Tag task instances	2848
9.12.4.5. Retroactive data generation instances	2850
9.13. Monitoring and alerting	2854
9.13.1. Alert records for batch processing tasks	2854
9.13.2. Alert rules for batch processing tasks	2857
9.13.3. Alert records for stream processing tasks	2859
9.13.4. Alert rules for stream processing tasks	2861
9.14. Data assets	2865
9.14.1. Overview	2865
9.14.2. Map	2866
9.14.2.1. View the asset map	2866
9.14.2.2. View table details	2868
9.14.3. Security	2872
9.14.3.1. My permissions	2872
9.14.3.1.1. Manage permissions on logical tables	2872
9.14.3.1.2. Manage permissions on physical tables	2874
9.14.3.1.3. Manage permissions on stream metatables	2876
9.14.3.1.4. Manage permissions on functions	2878
9.14.3.1.5. Manage permissions on data sources	2879
9.14.3.1.6. Manage permissions on API operations	2880
9.14.3.1.7. Manage functionality permissions	2881
9.14.3.2. Managed permissions	2882
9.14.3.2.1. Grant and revoke permissions on business units	2882
9.14.3.2.2. Grant and revoke permissions on projects	2883
9.14.3.2.3. Grant and revoke permissions on data sources	2884
9.14.3.2.4. Grant and revoke functionality permissions	2885

9.14.3.3. Owned permissions	2886
9.14.3.3.1. Configure functionality permissions and transfer... ..	2886
9.14.4. Data governance	2886
9.14.4.1. Overview	2886
9.14.4.2. Resource analysis	2887
9.14.4.3. Governance overview	2892
9.14.4.3.1. Governance analysis	2892
9.14.4.3.2. Governance results	2895
9.14.4.4. Governance console	2900
9.14.4.5. Artifact management	2904
9.14.4.5.1. Artifact management process	2904
9.14.4.5.2. Metadata registration	2905
9.14.4.5.3. Artifact management	2909
9.14.4.5.4. Push management	2914
9.14.4.5.5. Task management	2917
9.14.4.6. Recycle bin	2919
9.14.5. Data quality	2920
9.14.5.1. Overview	2920
9.14.5.2. Terms	2921
9.14.5.3. View statistics on the Overview page	2921
9.14.5.4. Manage quality rules	2923
9.14.5.4.1. View quality rules	2923
9.14.5.4.2. Configure a quality rule	2925
9.14.5.4.3. Modify a quality rule	2931
9.14.5.4.4. Delete a quality rule	2931
9.14.5.5. View check records	2932
9.14.5.6. View a quality report	2933
9.15. Theme-based data service	2936

9.15.1. Ad hoc query	2936
9.16. Data services	2938
9.16.1. Overview	2938
9.16.2. Terms	2940
9.16.3. Configure the network	2940
9.16.4. Add a member	2942
9.16.5. Develop API operations	2943
9.16.5.1. Create a group	2943
9.16.5.2. Create a service unit	2943
9.16.5.2.1. Create a metadata set	2943
9.16.5.2.2. Create a service unit from a physical table	2945
9.16.5.2.3. Create a service unit from multiple physical ta...	2947
9.16.5.3. Create an API operation	2950
9.16.5.3.1. Create an API operation in template wizard mo...	2950
9.16.5.3.2. Create an API operation in custom SQL mode	2952
9.16.5.4. Test an API operation	2955
9.16.5.5. Publish an API operation	2956
9.16.5.6. Unpublish an API operation	2957
9.16.5.7. Delete an API operation	2957
9.16.6. Use and manage an API operation	2958
9.16.6.1. Create an application	2958
9.16.6.2. View an API operation	2959
9.16.6.3. Apply for permissions on an API operation	2960
9.16.6.4. Debug an API operation	2961
9.16.6.5. Call an API operation	2961
9.16.7. Monitor an API operation	2962
9.16.8. Create a Dataphin data source	2964
9.16.9. Use and manage a Dataphin data source	2966

9.16.9.1. View a Dataphin data source	2966
9.16.9.2. Use a Dataphin data source	2966
10. Elasticsearch (on ECS)	2968
10.1. What is Apsara Stack Elasticsearch?	2968
10.2. Quick start	2968
10.2.1. Log on to the Elasticsearch console	2968
10.2.2. Create an Elasticsearch cluster	2969
10.2.3. Access an Elasticsearch cluster	2972
10.3. Manage clusters	2973
10.3.1. Log on to the Kibana console	2973
10.3.2. Restart an Elasticsearch cluster	2974
10.3.3. Refresh the information of an Elasticsearch cluster	2974
10.3.4. View the basic information of an Elasticsearch cluster	2975
10.3.5. Upgrade the configuration of an Elasticsearch cluster	2976
10.3.6. Configure an Elasticsearch cluster	2976
10.3.6.1. Configure synonyms	2976
10.3.6.2. Perform configurations on YAML files	2986
10.3.6.2.1. Configure a YAML file	2986
10.3.6.2.2. YAML configuration parameters	2987
10.3.6.2.3. Reindex data from a remote Elasticsearch clust...	2989
10.3.7. Configure plug-ins	2991
10.3.8. Configure security settings	2994
10.3.9. Back up data	2995
10.3.9.1. Enable and configure the auto snapshot feature	2995
10.3.9.2. Query snapshot status	2996
10.3.9.3. Restore data from automatic snapshots	2999
10.3.9.4. Commands for creating snapshots and restoring d...	3001
10.4. Manage documents	3010

10.4.1. Create a document	3010
10.4.2. Update a document	3011
10.4.3. Retrieve a document	3011
10.4.4. Search for documents	3012
10.4.5. Perform a complex search	3012
10.4.6. Delete a document	3013
10.5. Elasticsearch test	3014
10.5.1. Use a curl command to access an Elasticsearch cluster...	3014
10.5.2. Use Python to access an Elasticsearch cluster over po...	3014
10.5.3. Use Java REST Client to access an Elasticsearch cluste...	3015
11.Elasticsearch (on k8s)	3017
11.1. What is Apsara Stack Elasticsearch?	3017
11.2. Quick start	3017
11.2.1. Log on to the Elasticsearch console	3017
11.2.2. Access an Elasticsearch cluster	3018
11.2.3. View the information of an Elasticsearch cluster	3019
11.2.4. Create an index	3020
11.2.5. Manage documents	3021
11.2.5.1. Create a document	3021
11.2.5.2. Update a document	3023
11.2.5.3. Retrieve a document	3024
11.2.5.4. Search for documents	3025
11.2.5.5. Perform a complex search	3026
11.2.5.6. Perform statistical analytics	3029
11.2.5.7. Delete a document	3031
11.2.6. Delete an index	3032
11.3. Manage clusters	3032
11.3.1. Customize a cluster list	3032

11.3.2. Export a cluster list	3033
11.3.3. Refresh the information of a cluster	3033
11.3.4. View the basic information of an Elasticsearch cluster	3033
11.3.5. Log on to the Kibana console	3034
11.3.6. Create a snapshot and restore data	3035
11.4. Use plug-ins	3043
11.4.1. Use the a-pack-xdcr plug-in to replicate data across E... ..	3043
11.4.2. Use the opendistro_sql plug-in to query cluster data	3045
11.4.3. Use the bsearch_querybuilder plug-in to construct qu... ..	3046
11.4.4. Use alerting plug-ins to implement alerting for an Ela... ..	3047

1. MaxCompute

1.1. What is MaxCompute

MaxCompute is a data processing platform developed by Alibaba Group to process large amounts of data. MaxCompute provides channels for data uploads and downloads, a range of computing and analysis features including MaxCompute SQL and MaxCompute MapReduce, and comprehensive security solutions.

MaxCompute is used to store and compute large amounts of structured data. It provides warehousing solutions for large amounts of data, as well as big data analytics and modeling services.

As data collection techniques are becoming increasingly diverse and comprehensive, industries are amassing larger amounts of data. The data amount has increased from 100 GB, 1 TB to even 1 PB, far exceeding the processing capabilities of traditional software. Analysis tasks for large amounts of data require distributed computing instead of reliance on a single server. However, distributed computing models require skilled data analysts and are difficult to maintain. To use a distributed computing model, data analysts must understand business requirements and underlying computing models.

MaxCompute aims to provide easy analysis and processing of large amounts of data. You can analyze big data without a deep knowledge of distributed computing. MaxCompute is widely implemented within Alibaba Group for multiple scenarios. The scenarios include data warehousing and BI analysis for large Internet enterprises, website log analysis, e-commerce transaction analysis, and exploration of user characteristics and interests.

MaxCompute provides the following features:

- Data channels
 - Tunnel: provides highly concurrent uploads and downloads of offline data. MaxCompute Tunnel enables you to upload or download large amounts of data to or from MaxCompute. MaxCompute Tunnel provides a Java API.
 - DataHub: provides real-time data uploads and downloads. Data uploaded by using DataHub is available immediately, while data uploaded by using MaxCompute Tunnel is not.
- Computing and analysis
 - SQL: MaxCompute stores data in tables and provides SQL query capabilities. MaxCompute can be used as traditional database software, but it is capable of processing terabytes and petabytes of data. MaxCompute SQL does not support transactions, indexes, or operations such as UPDATE and DELETE. The SQL syntax used in MaxCompute is different from that in Oracle and MySQL. SQL statements from other database engines cannot be seamlessly migrated to MaxCompute. MaxCompute SQL responds to queries within a few minutes or seconds, instead of milliseconds. MaxCompute SQL is easy to learn. You can get started with MaxCompute SQL based on your prior experience of database operations, without a deep knowledge of distributed computing.
 - MapReduce: First proposed by Google, MapReduce is a distributed data processing model that has gained extensive attention and been used in a wide range of business scenarios. This document briefly describes the MapReduce model. You must have a basic knowledge of distributed computing and relevant programming experience before you use MapReduce. MapReduce provides a Java API.
 - Graph: an iterative graph computing framework provided in MaxCompute. Graph computing jobs use graphs to build models. A graph is a collection of vertices and edges that have values. MaxCompute Graph iteratively edits and evolves graphs to obtain analysis results.

- Unstructured data access and processing in integrated computing scenarios: MaxCompute SQL cannot directly process external data, such as unstructured data from Object Storage Service (OSS). Data must be imported to MaxCompute tables by using relevant tools before computation. The MaxCompute team introduces the unstructured data processing framework to the MaxCompute system architecture to handle this issue.
MaxCompute allows you to create external tables to process data from the following data sources:
 - Internal data sources: OSS, Tablestore, AnalyticDB, ApsaraDB RDS, Alibaba Cloud HDFS, and TDDL
 - External data sources: open source HDFS, MongoDB, and HBase
- Unstructured data access and processing in MaxCompute: MaxCompute reads and writes volumes to store and process unstructured data, which otherwise must be stored in an external storage system.
- Spark on MaxCompute: a big data analytics engine developed by Alibaba Cloud to provide big data processing capabilities for Alibaba Group, government agencies, and enterprises. For more information, see [Spark on MaxCompute](#).
- SDK: a toolkit provided for developers.
- Security solution: MaxCompute provides powerful security features to ensure data security.

1.2. Usage notes

You can selectively read topics in this document based on your requirements. This topic provides reading recommendations based on your roles.

MaxCompute beginners

If you are a beginner in MaxCompute, we recommend that you first familiarize yourself with the following topics:

- What is MaxCompute: This topic provides the MaxCompute overview and describes its features. It helps you obtain a general knowledge of MaxCompute.
- Quick start: This topic describes how to download and configure the client, create a table, grant permissions, import data, and export data. It also describes how to run SQL jobs, user-defined functions (UDFs), and MapReduce programs.
- Terms and common statements: This topic introduces the basic terms of MaxCompute and commonly used statements in MaxCompute. It helps you familiarize yourself with operations on MaxCompute.
- Frequently used tools: This topic describes how to download, configure, and use the commonly used tools in MaxCompute before you analyze data.

Data analysts

If you are a data analyst, we recommend that you familiarize yourself with the following topics:

MaxCompute SQL: The topic describes how to query and analyze large amounts of data stored in MaxCompute.

MaxCompute SQL provides the following features:

- Allows you to use the CREATE, DROP, and ALTER DDL statements to manage tables and partitions.
- Allows you to execute the following DML statements:
 - You can use the SELECT statement to select data records in a table and the WHERE clause to query data records that meet specific conditions. These statements help you search for data records.
 - You can join two tables by using equi-joins.
 - You can use the GROUP BY clause to aggregate columns.
 - You can use the INSERT OVERWRITE or INSERT INTO statement to insert data records into another table.

- You can use SELECT TRANSFORM to simplify the reference of script code.
- Allows you to use built-in functions and UDFs to complete a variety of computations.
- Allows you to use user-defined types (UDTs) to reference classes or objects of third-party programming languages in SQL statements and therefore to obtain data or call methods.
- Allows you to use UJEs to implement flexible cross-table and multi-table custom operations and reduce the detailed operations on the underlying distributed systems by using methods such as MapReduce.
- Allows you to collect statistics on tables and configure table lifecycles.
- Supports regular expressions.

Users with development experience

If you have development experience, understand the distributed architecture, and want to obtain data analytics capabilities that SQL cannot deliver, we recommend that you read the following advanced functional topics of MaxCompute:

- MaxCompute MapReduce: a MapReduce programming model for Java. You can use the Java API provided by MapReduce to write MapReduce programs and process MaxCompute data.
- MaxCompute Graph: a processing framework for iterative graph computing. A graph consists of vertices and edges, both of which contain values. MaxCompute Graph iteratively edits and evolves graphs to obtain analysis results.
- Eclipse plug-in: an integrated development environment (IDE) to help you complete development by using MapReduce, UDFs, and Graph SDK for Java.
- SDK for Java: a toolkit provided for developers.
- MaxCompute Tunnel: enables you to upload or download large amounts of data to or from MaxCompute at a time.

Project owners or administrators

If you are a project owner or an administrator, we recommend that you familiarize yourself with the following topics:

Security solution: This topic describes how to authorize users, enable cross-project resource sharing, configure project protection, and perform policy-based authorization.

1.3. Preparations

1.3.1. Log on to the Apsara Uni-manager Management Console

This topic describes how to log on to the Apsara Uni-manager Management Console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

Note When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.

1.3.2. Create an Apsara Stack tenant account

This topic describes how to create an Apsara Stack tenant account.

Procedure

1. Log on to the Apsara Uni-manager Management Console as an administrator.
2. In the top navigation bar, choose **Products > Big Data > MaxCompute** to go to the **MaxCompute** homepage.
3. In the left-side navigation pane of the page that appears, click **Task Accounts**. In the upper-right corner of the page that appears, click **Create Task Account**.
4. In the dialog box that appears, configure the parameters and click **OK**.

Parameter	Description
Name	The name of the Apsara Stack tenant account.
Organization	The organization to which the Apsara Stack tenant account belongs.
Description	The description of the Apsara Stack tenant account. You can leave this parameter empty.

1.3.3. Create a project

This topic describes how to create a MaxCompute project.

Procedure

1. Log on to the Apsara Uni-manager Management Console as an administrator.
2. In the top navigation bar, choose **Products > Big Data > MaxCompute** to go to the **MaxCompute** homepage.
3. In the left-side navigation pane, click **Projects**. In the upper-right corner of the page that appears, click **Create MaxCompute Project**.
4. On the page that appears, configure the parameters and click **Submit**. Configurations in the Region section

Parameter	Description
Organization	The organization that you select for the project.
Resource Set	The resource set of the selected organization.
Cross-domain Enhancement Pack	The region with which the project you create can associate.
VPC	The virtual private cloud (VPC) in the region. The VPC is used to isolate networks.
Cluster	The information about the cluster of the project.

Configurations in the Security Enhancement Switch section

Parameter	Description
Enhancement Pack for Joint Computing	This switch is turned off by default. You can turn on the switch to support joint computing features.
Encrypted	Specifies whether to encrypt data. Default value: No.
Encryption Method	This parameter is not displayed by default. It is displayed when you set Encrypted to Yes. Valid values: AESCTR, AES256, RC4, and SM4.
Encryption Key	This parameter is not displayed by default. It is displayed when you set Encrypted to Yes. Default value: Generate Automatically.

Configurations in the Resource Configurations section

Parameter	Description
Quota Group	The quota group that corresponds to the cluster. If no groups are available, click Create Quota Group to create one. For more information about the configurations, see Create a quota group .
Storage	You need only to specify the Requested (GB) parameter. The Requested (GB) parameter specifies the storage that is requested for the project. The value cannot be greater than the total capacity of the disk.

Configurations in the Basic Settings section

Parameter	Description
Project Name	The name of the project you want to create.
Owner Account	The name of the Apsara Stack tenant account that corresponds to the project owner. The value of this parameter depends on the associated organization. You cannot specify this parameter.

Parameter	Description
Task Account	Optional. The account of the task. If no accounts are available, click Create Task Account to create one. For more information about the creation process and detailed configurations, see Prepare an Apsara Stack tenant account .
Description	The description of the project. This parameter can be empty.

1.3.4. Create a quota group

This topic describes how to create a quota group in MaxCompute.

Procedure

1. Log on to the Apsara Uni-manager Management Console as an administrator.
2. In the top navigation bar, choose **Products > Big Data > MaxCompute** to go to the **MaxCompute** homepage.
3. In the left-side navigation pane, click **Quota Groups**. In the upper-right corner of the page that appears, click **Create**.
4. On the page that appears, configure the parameters and click **Submit**. Configurations in the Region section

Parameter	Description
Organization	The organization that you select for the quota group.
Resource Set	The resource set in the selected organization.
Region	The region of the cluster where the quota group resides.
Cluster	The information about the cluster of the quota group.

Configurations in the Basic Settings section

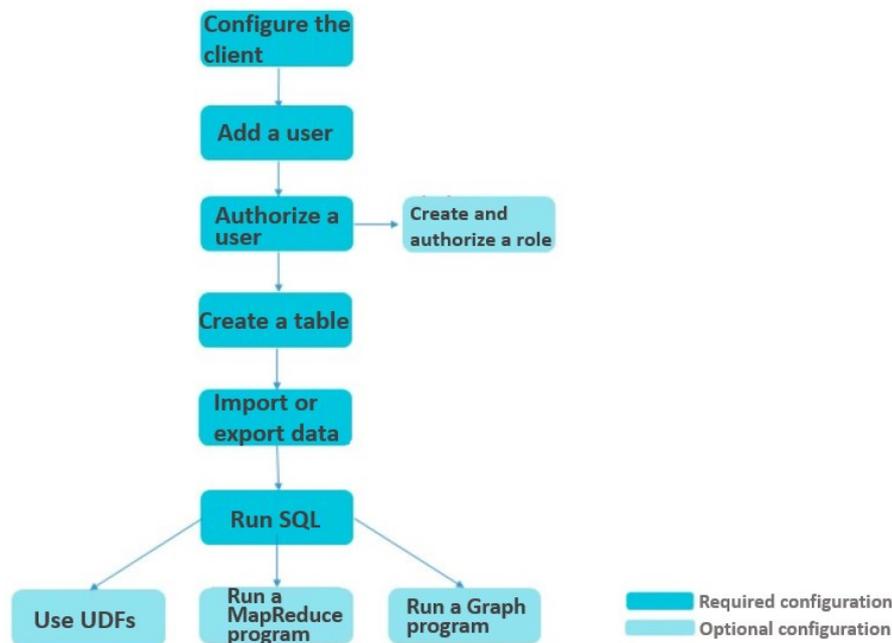
Parameter	Description
Quota Group Name	The name of the quota group that you want to create.
Sharing Scope	The sharing scope of the quota group. Valid values: Current Resource Set, Current Organization and Subordinate Organizations, and Current Organization.
Quantity	The number of resources to allocate to the current quota group. The value must be greater than 0.

1.4. Quick start

1.4.1. Overview

This topic describes the procedure to use MaxCompute. It provides you with step-by-step instructions on basic MaxCompute operations.

The following figure shows the procedure to use MaxCompute.



1. **Configure the client.**

You must install and configure the client to access MaxCompute projects and use all the features of MaxCompute. For more information, see [MaxCompute client](#).

2. **Add a user.**

Except for the project owner, all users must be manually added to a project and granted permissions before they can perform operations on the project.

3. **Authorize a user.**

After you add a user, you must authorize the user to perform operations on the project. A user can perform operations on the project only after the user is authorized.

4. (Optional) **Create and authorize a role.**

It is time-consuming to authorize each user in a project that contains a large number of users. Project administrators can use roles to grant users a specified set of permissions. After you authorize a role, all users who assume this role are granted the same permissions.

5. **Create a table.**

After a user is added to a project and authorized, the user can use MaxCompute. Table operations are the most basic operations in MaxCompute.

6. **Import or export data.**

You can use the SDK provided by MaxCompute Tunnel to write your own Java programs and import or export data.

7. **Execute SQL statements.**

You can familiarize yourself with the limits on a few common SQL statements. For more information about how to execute SQL statements, see [MaxCompute SQL](#).

8. Perform the following optional operations:

- [Use UDFs](#).

After you install the MaxCompute client, you can try to use user-defined functions (UDFs). MaxCompute supports user-defined scalar functions, user-defined aggregate functions (UDAFs), and user-defined table-valued functions (UDTFs).

- [Run a MapReduce program](#).

After you install the MaxCompute client, you can run a MapReduce program.

- [Run a Graph program](#).

After you install the MaxCompute client, you can run a Graph program.

1.4.2. Configure the MaxCompute client

The MaxCompute client allows you to access MaxCompute projects and use MaxCompute features. This topic describes how to download, configure, and run the client.

Prerequisites

JRE 1.8 is installed on your on-premises machine because the MaxCompute client is developed in Java. In addition, you have an Apsara Stack tenant account and have obtained AccessKey ID and AccessKey secret.

 **Note** Before you configure the MaxCompute client, make sure that a project is created and AccessKey ID and AccessKey secret are obtained.

Procedure

1. Download the [client](#) package to your on-premises machine.
2. Decompress the package to the folder in which you want to store the client. The package contains the following folders:

```
bin/  
conf/  
lib/  
plugins/
```

3. Edit the `odps_config.ini` file in the `conf` folder. The following code shows the configuration information:

```
project_name=my_project  
access_id=*****  
access_key=*****  
end_point= <Endpoint of MaxCompute>
```

- `access_id` and `access_key` are the AccessKey pair of the Apsara Stack tenant account. `access_id` corresponds to AccessKey ID and `access_key` corresponds to AccessKey secret.
 - `project_name=my_project` specifies the project that you want to access. This is the default project that is accessed each time you log on to the client. If this parameter is not specified, you must run the `use project_name` command to access the project after you log on to the client.
 - Set `end_point` to the endpoint of MaxCompute. The endpoint varies based on the user account.
4. After the modification, run the `odpscmd` file in Linux or the `odpscmd.bat` file in Windows in the `bin` directory to execute SQL statements and related commands. Example:

```
create table tbl1(id bigint);
insert overwrite table tbl1 select count(*) from tbl1;
select 'welcome to MaxCompute!' from tbl1;
```

 **Note** For more information about SQL statements, see [MaxCompute SQL](#).

1.4.3. Add and delete a user

Except for a project owner, all users must be manually added to the project and granted permissions before they can perform operations on the project. This topic describes how to add users to or delete users from a project as a project owner.

If you are a project owner, we recommend that you read this topic in full. If you are a regular user, we recommend that you submit an application to a project owner to join a project, and read the subsequent topics after you are added to the project.

Add a user

Syntax:

```
add user <full_username>;
```

Examples:

```
add user bob@aliyun.com;
```

If you are not sure whether the user is already in the project, run the following command to query the users in the project:

```
list users;
```

 **Note**

- After a user is added to a MaxCompute project, the user must be authorized by the project owner. Then, the user can perform authorized operations on the project.
- For more information about authorization, see [Grant and view permissions](#).

Delete a user

Syntax:

```
remove user <full_username>;
```

Examples:

```
remove user bob@aliyun.com;
```

 Note

- Before you delete a user, make sure that you have revoked all the permissions from the user. If you delete a user before you revoke the permissions from the user, the permissions are retained. If the user is added to the project again, the user still has the permissions that were previously granted.
- For more information about how to add or delete users, see [User management](#).

1.4.4. Grant and view permissions

1.4.4.1. Overview

After you add a user, you must authorize the user to perform operations on the project. A user can perform operations on the project only after the user is authorized.

Authorization is a process of granting permissions to perform operations on objects, such as tables, tasks, and resources, in MaxCompute. The permissions include read, write, and view permissions.

The following topics are intended for project administrators. If you are a regular MaxCompute user, make sure that you have obtained the required permissions.

MaxCompute provides two authorization mechanisms. For more information, see [ACL authorization](#) and [Policy authorization](#).

1.4.4.2. ACL-based authorization

This topic describes the statements for ACL-based authorization and provides examples.

ACL-based authorization in MaxCompute applies to the following objects: projects, tables, functions, resources, instances, and tasks. Each type of object requires different operation permissions. For more information, see [ACL-based authorization actions](#).

Syntax:

```
grant actions on object to subject
revoke actions on object from subject
actions ::= action_item1, action_item2, ...
object ::= project project_name | table schema_name |
         instance inst_name | function func_name |
         resource res_name
subject ::= user full_username | role role_name
```

Examples:

alice@aliyun.com is newly added to the test_project_a project. Allen is a RAM user of bob@aliyun.com. An Alibaba Cloud account can execute the following statements to grant permissions, including the permissions to submit jobs, create data tables, and query existing objects in a project:

```
-- Enter a project.
use test_project_a;
-- Add a member to the project.
add user aliyun$alice@aliyun.com;
-- Add a RAM user.
add user ram$bob@aliyun.com:Allen;
-- Create a role named worker.
create role worker;
-- Assign the worker role to the added members.
grant worker TO aliyun$alice@aliyun.com;
grant worker TO ram$bob@aliyun.com:Allen;
-- Grant the CREATE INSTANCE, CREATE RESOURCE, CREATE FUNCTION, CREATE TABLE, and LIST permissions to the
worker role.
grant CreateInstance, CreateResource, CreateFunction, CreateTable, List ON PROJECT test_project_a TO ROLE wor
ker;
-- Grant all instance permissions to the worker role.
grant all on instance instance_name to Role worker;
```

1.4.4.3. Policy-based access control

This topic describes the commands that you can use for policy-based access control and provides examples on how to configure a policy.

Policy-based access control is subject-based authorization. For more information, see [Authorization policies](#).

Commands:

```
GET POLICY;
-- Read the project policy.
PUT POLICY <policyFile>;
-- Specify or overwrite the project policy.
GET POLICY ON ROLE <roleName>;
-- Read the policy of a role in the project.
PUT POLICY <policyFile> ON ROLE <roleName>;
-- Specify or overwrite the policy of a role in the project.
```

Example on how to configure a policy:

```
{
  "Version": "1",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "odps:*"
      ],
      "Resource": "acs:odps*:projects/$user_project_name/tables/*",
      "Condition": {
        "StringEquals": {
          "odps:TaskType": [
            "DT"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "odps:List",
        "odps:Read",
        "odps:Describe",
        "odps:Select"
      ],
      "Resource": [
        "acs:odps*:projects/$user_project_name/tables/a",
        "acs:odps*:projects/$user_project_name"
      ],
      "Condition": {
        "StringEquals": {
          "odps:TaskType": [
            "SQL"
          ]
        }
      }
    }
  ]
}
```

 **Note** The preceding example disables the Tunnel feature of \$user_project_name, and grants the user the LIST, READ, DESCRIBE, and SELECT permissions only on the project and table a in the project.

1.4.4.4. Query permissions

You can execute a statement to query user permissions in MaxCompute.

Execute the following statement to query the permissions of a user:

```
show grants for $user_name;
```

 **Note** For more information, see [View permissions](#).

1.4.5. Create and authorize a role

If a project has a large number of users, the authorization process is time-consuming. Project administrators can use roles to grant users a specified set of permissions. After you authorize a role, all users who assume this role are granted the same permissions. This topic describes how to create and authorize a role.

One user can assume multiple roles, and multiple users can assume the same role.

Create a role

Syntax:

```
create role <roleName>;
```

Examples:

```
create role player;
```

Assign a role to a user

Syntax:

```
grant <roleName> to <full_username>;
```

Examples:

```
grant player to bob@aliyun.com;
```

Delete a role

Syntax:

```
drop role <roleName>;
```

Examples:

```
drop role player;
```

 **Note** Before you delete a role, make sure that all users have been removed from the role.

Authorize a role

Role authorization is similar to user authorization. For more information about how to authorize roles, see [User authorization](#). For more information about role authorization, see [Role management](#).

1.4.6. Create or delete a table

1.4.6.1. Create a table

This topic describes how to create a table.

Syntax:

```
create table [if not exists] table_name
[(col_name data_type [comment col_comment], ...)] [comment table_comment]
[partitioned by (col_name data_type [comment col_comment], ...)] [lifecycle days]
[as select_statement]
-- You can also execute the following statement to create a table:
create table [if not exists] table_name
like existing_table_name
```

Examples:

```
create table test1 (key string);
-- Create a non-partitioned table.
create table test2 (key bigint) partitioned by (pt string, ds string);
-- Create a partitioned table.
create table test3 (key boolean) partitioned by (pt string, ds string) lifecycle 100;
-- Create a table that has a lifecycle specified.
create table test4 like test3;
-- Create a table named test4 that has the same attributes, such as the field type and partition type, as those of test 3. The only difference between the two tables is the lifecycle.
create table test5 as select * from test2;
-- Create a table named test5. However, the partitioning and lifecycle attributes are not replicated to the destination table. Only data of test2 is replicated to test5.
```

You can specify partitions and lifecycles for MaxCompute tables. For more information about how to create a table, see [Create a table](#). For more information about how to modify partitions, see [Add a partition](#) and [Delete a partition](#). For more information about how to modify lifecycle settings, see [Modify the lifecycle of a table](#).

1.4.6.2. Query table information

This topic describes how to query table information.

Syntax:

```
desc <table_name>;
```

Examples:

```
desc test3;
-- Query information about test3.
desc test4;
-- Query information about test4.
```

1.4.6.3. Delete a table

This topic describes how to delete a table.

Syntax:

```
drop table [if exists] table_name;
```

Examples:

```
drop table test2;
```

 **Note** For more information, see [Delete a table](#).

1.4.7. Import or export data

This topic describes how to compile Java programs by using the MaxCompute Tunnel SDK and import data to or export data from MaxCompute.

For more information about data import and export, see [Tunnel SDK example](#).

1.4.8. Run SQL

1.4.8.1. Overview

This topic describes the limits on the execution of SQL statements.

For more information about how to execute SQL statements, see [MaxCompute SQL](#).

Take note of the following points when you use MaxCompute SQL:

- MaxCompute SQL does not support transactions, indexes, or operations such as UPDATE and DELETE.
- The SQL syntax of MaxCompute is different from that of Oracle or MySQL. You cannot directly migrate SQL statements from other database engines to MaxCompute.
- MaxCompute SQL does not respond to queries in real time. It requires a few minutes to return query results, instead of seconds or milliseconds.

1.4.8.2. SELECT

This topic describes limits of the SELECT statements.

The following limits apply to the SELECT statements:

- The key of the GROUP BY clause can be the name of a column in the input table, or the expression composed of columns in the input table. However, it cannot be the output column from the SELECT operation.

```
select substr(col2, 2) from tbl group by substr(col2, 2);  
-- This statement can be executed because the key of the GROUP BY clause is the expression composed of columns in the input table.  
select col2 from tbl group by substr(col2, 2);  
-- This statement cannot be executed because the key of the GROUP BY clause is not included in the columns of the SELECT operation.  
select substr(col2, 2) as c from tbl group by c;  
-- This statement cannot be executed because the key of the GROUP BY clause is the alias of an output column from the SELECT operation.
```

 **Note** In most cases, the GROUP BY clause precedes the SELECT operation during the parsing of SQL statements. Therefore, GROUP BY uses only the column names in the input table or expressions of columns as keys.

- The DISTRIBUTE BY clause must be placed before the SORT BY clause.
- The key of the ORDER BY, SORT BY, or DISTRIBUTE BY clause must be the alias of an output column from the SELECT operation.

```
select col2 as c from tbl order by col2 limit 100;
-- This statement cannot be executed because the key of the ORDER BY clause is not the alias of an output column from the SELECT operation.
select col2 from tbl order by col2 limit 100;
-- This statement can be executed. If an output column from the SELECT operation does not have an alias, the column name is used as the alias.
```

 **Note** In most cases, the SELECT operation is performed before the ORDER BY, SORT BY, and DISTRIBUTE BY clauses during the parsing of SQL statements. Therefore, only the aliases of output columns from the SELECT operation can be used as keys of these clauses.

For more information about the SELECT statements, see [SELECT](#).

1.4.8.3. INSERT

This topic describes the limits of the INSERT statements.

The following limits apply to the INSERT statements:

- If you execute an INSERT statement to insert data into a partition, the partition key columns cannot be included in the SELECT clause.

```
insert overwrite table sale_detail_insert partition (sale_date='2017', region='china') select shop_name, customer_id, total_price, sale_date, region from sale_detail;
-- An error is returned. The sale_date and region columns are partition key columns and cannot be included in the INSERT statement that is used to insert data into static partitions.
```

- If you execute an INSERT statement to insert data into dynamic partitions, the dynamic partition key columns must be included in the SELECT clause.

```
insert overwrite table sale_detail_dypart partition (sale_date='2017', region) select shop_name, customer_id, total_price from sale_detail;
-- An error is returned. If you execute an INSERT statement to insert data into dynamic partitions, the dynamic partition key columns must be included in the SELECT clause.
```

For more information about the INSERT statements, see [INSERT](#).

1.4.8.4. JOIN

This topic describes the limits on JOIN statements.

Limits:

- MaxCompute SQL supports the following JOIN operations: {LEFT OUTER|RIGHT OUTER|FULL > OUTER|INNER} JOIN.
- MaxCompute SQL supports a maximum of 128 concurrent JOIN operations.

For more information, see [JOIN](#).

1.4.8.5. Other limits

This topic describes other limits on MaxCompute SQL.

- MaxCompute SQL supports a maximum of 256 concurrent UNION operations.
- MaxCompute SQL supports a maximum of 256 concurrent INSERT OVERWRITE or INSERT INTO operations.

1.4.9. Compile and use UDFs

1.4.9.1. Overview

This topic describes how to write and run a MaxCompute user-defined function (UDF).

MaxCompute supports user-defined scalar functions, user-defined aggregate functions (UDAFs), and user-defined table-valued functions (UDTFs).

Note

- UDFs support only Java APIs. To write a UDF program, you can upload UDF code to your project as a resource and execute the required statement to create a UDF.
- This topic provides code examples of user-defined scalar functions, UDAFs, and UDTFs.

1.4.9.2. UDF example

This topic describes how to create a user-defined scalar function (UDF) and provides an example.

Procedure

- Write code. You must develop and compile functions in compliance with the MaxCompute UDF framework.

```
package org.alidata.odps.udf.examples; import com.aliyun.odps.udf.udf;
public final class Lower extends udf { public String evaluate(String s) {
if (s == null) { return null; } return s.toLowerCase();
}
}
```

Name the preceding JAR package *my_lower.jar*.

- Add resources. Specify the referenced UDF code before you run a UDF. User code must be added to MaxCompute as resources. Java UDFs must be packaged as JAR packages and added to MaxCompute as JAR resources. The UDF framework automatically loads the JAR packages and runs the UDFs.

Example command to add JAR resources:

```
add jar my_lower.jar;
```

 **Note** If multiple resources have the same name, rename the JAR packages and modify the names of relevant JAR packages in the example command. You can also use the `-f` option to override the existing JAR resources.

- Register the UDF. After your JAR package is uploaded, MaxCompute has no information about this UDF. Therefore, you must register a unique function name in MaxCompute and associate the function name with the JAR resource and function.

Example command to register the UDF:

```
create function test_lower as org.alidata.odps.udf.examples.lower using my_lower.jar;
```

Example of the function used in SQL:

```
select test_lower('A') from my_test_table;
```

1.4.9.3. UDAF example

This topic provides the code example of a user-defined aggregate function (UDAF) for your reference.

UDAFs are registered in the same way as UDFs and are used in the same way as built-in aggregate functions.

UDAF code example:

```
package org.alidata.odps.udf.examples;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.io.Writable;
import com.aliyun.odps.udf.Aggregator;
import com.aliyun.odps.udf.UDFException;
/**
 * project: example_project
 * table: wc_in2
 * partitions: p2=1,p1=2
 * columns: colc,colb,cola
 */
public class UDAFExample extends Aggregator {
    @Override
    public void iterate(Writable arg0, Writable[] arg1) throws UDFException {
        LongWritable result = (LongWritable) arg0;
        for (Writable item : arg1) {
            Text txt = (Text) item;
            result.set(result.get() + txt.getLength());
        }
    }
    @Override
    public void merge(Writable arg0, Writable arg1) throws UDFException {
        LongWritable result = (LongWritable) arg0;
        LongWritable partial = (LongWritable) arg1;
        result.set(result.get() + partial.get());
    }
    @Override
    public Writable newBuffer() {
        return new LongWritable(0L);
    }
    @Override
    public Writable terminate(Writable arg0) throws UDFException {
        return arg0;
    }
}
```

1.4.9.4. UDTF example

This topic provides an example of user-defined table-valued function (UDTF) code.

UDTFs are registered and used in the similar way to UDFs.

UDTF code example:

```

package org.alidata.odps.udtf.examples;
import com.aliyun.odps.udf.udtf;
import com.aliyun.odps.udf.udtfcollector; import com.aliyun.odps.udf.annotation.resolve; import com.aliyun.odps.udf.udfexception;
// todo define input and output types, e.g., "string,string->string,bigint".
@resolve({"string,bigint->string,bigint"}) public class myudtf extends udtf {
@override
public void process(object[] args) throws udfexception { string a = (string) args[0];
long b = (long) args[1];
for (string t: a.split("\\s+")) { forward(t, b);
}
}
}
}

```

1.4.10. Write and run a MapReduce program

This topic describes how to write and run a MapReduce program.

Prerequisites

- JDK 1.8 is installed.
- The MaxCompute client is configured. For more information, see [Configure the client](#).

Procedure

1. Create input and output tables.Example:

```

create table wc_in (key string, value string);
create table wc_out (key string, cnt bigint);

```

 **Note** For more information about the statements that are used to create tables, see [Create a table](#).

2. Insert data into the wc_in table.

You can use one of the following methods to insert data:

- Run Tunnel commands to upload data.

The following code shows the data that you want to insert into the table. You must create a kv.txt file on your computer and save the data to the file. In this example, the kv.txt file is saved in D:\.

```

238,val_238
186,val_86
186,val_86

```

Run the following command to upload the data:

```
Tunnel upload D:\kv.txt wc_in;
```

- Execute the following SQL statement to insert the data:

```
INSERT INTO TABLE wc_in VALUES ('238','val_238'),('186','val_86'),('186','val_86');
```

3. Write a MaxCompute program and compile it.

- MaxCompute provides the Eclipse plug-in to help you develop MapReduce programs and debug them on your computer.

- You must create a MaxCompute project in Eclipse. Then, write a MapReduce program in this project. After local debugging succeeds, upload the compiled program (JAR package) to MaxCompute.
4. Add the JAR package as a project resource.

In this example, the JAR package is named `word-count-1.0.jar`.

```
add jar word-count-1.0.jar;
```

5. Run the JAR command on the MaxCompute client.

```
jar -resources word-count-1.0.jar -classpath /home/resources/word-count-1.0.jar com.taobao.jingfan.wordcount wc_in wc_out;
```

 **Note** If the Java program uses resources, use the `-resources` option to specify the resources.

6. View the command output on the MaxCompute client.

```
select * from wc_out;
```

1.4.11. Write and run a Graph job

This topic describes how to write and run a Graph job.

Procedure

1. Create input and output tables. Execute the following statements:

```
create table sssp_in (v bigint, es string);  
create table sssp_out (v bigint, l bigint);
```

 **Note** For more information about the statements that are used to create tables, see [Create a table](#).

2. Upload data to the `sssp_in` table.

We recommend that you create an `sssp.txt` file that contains the following data and save the file to a local directory, such as `D:\`:

```
1 2:2,3:1,4:4  
2 1:2,3:2,4:1  
3 1:1,2:2,5:1  
4 1:4,2:1,5:1  
5 3:1,4:1
```

Run the Tunnel command to upload data to the `sssp_in` table. Use spaces as the delimiter between columns.

```
tunnel u -fd " " D:\sssp.txt sssp_in;
```

3. Compile a Single Source Shortest Path (SSSP) example.

Compile and debug SSSP algorithm examples in local mode based on the Graph development process. For more information, see [SSSP](#).

 **Note** In this example, the code is packaged as `odps-graph-example-sssp.jar`. You need only to package the SSSP code. You do not need to package the SDK into this JAR package.

4. Add a JAR resource.Examples:

```
add jar $LOCAL_JAR_PATH/odps-graph-example-sssp.jar odps-graph-example-sssp.jar
```

 **Note** For more information about how to add resources, see [Add a resource](#).

5. Run the SSSP.Examples:

```
jar -libjars odps-graph-example-sssp.jar -classpath $LOCAL_JAR_PATH/odps-graph-example-sssp.jar com.aliyun.odps.graph.examples.sssp 1 sssp_in sssp_out;
```

 **Note**

The MaxCompute client provides a jar command to run MaxCompute Graph jobs. You can use this command for Graph jobs in the same way as MapReduce jobs.

The commands used to run Graph jobs list the instance ID, execution progress, and result summary of the job. Example:

```
ID = 20130730160742915g*****
2013-07-31 00:18:36 SUCCESS
Summary:
Graph Input/Output
Total input bytes=211
Total input records=5
Total output bytes=161
Total output records=5
graph_input_[bsp.sssp_in]_bytes=211
graph_input_[bsp.sssp_in]_records=5
graph_output_[bsp.sssp_out]_bytes=161
graph_output_[bsp.sssp_out]_records=5
Graph Statistics
Total edges=14
Total halted vertices=5
Total sent messages=28
Total supersteps=4
Total vertices=5
Total workers=1
Graph Timers
Average superstep time (milliseconds)=7
Load time (milliseconds)=8
Max superstep time (milliseconds) =14
Max time superstep=0
Min superstep time (milliseconds)=5
Min time superstep=2
Setup time (milliseconds)=277
Shutdown time (milliseconds)=20
Total superstep time (milliseconds)=30
Total time (milliseconds)=344
OK
```

1.4.12. Use Logview to view job running information

This topic describes how to use Logview to view job running information.

Logview is a tool that you can use to view and debug jobs after you submit the jobs to MaxCompute. You can use Logview to view the following information of a job:

- Running status of tasks
- Running results of tasks
- Details of each task and the progress of each step

 **Note** When you deploy MaxCompute, Logview is deployed by default.

After you submit a job to MaxCompute, the system generates a Logview URL. You can enter the Logview URL in the search engine and press Enter to view job information.

```
odps@ test_workshop001>select * from result_test;

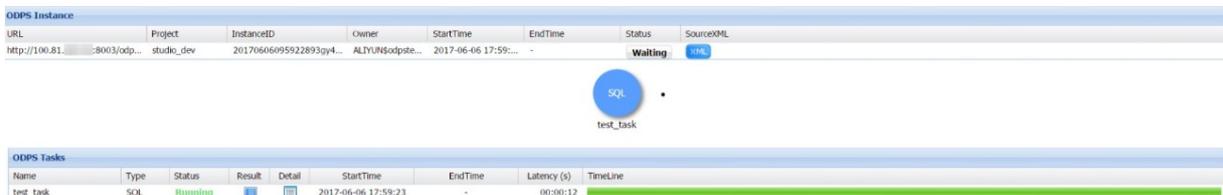
ID = 20190917055240226ga6e5mim
Log view:
http://logview.odps.aliyun.com/logview/?h=http://service.cn.maxcompute/api&p=test_workshop001&i=20190917055240226ga6e5mim&token=YkxhUzJQ1NIamRWnDBBU3UJPSxPRFBTX09CTzoxMDc50TI20Dk20Tk5NDIxLDE1NjkzMDQzNjAsnQi01t7IkFjdG1ubiI6WyJvZHBz01JlYWQiXSwiRWZmZWNOIjoiQWxsY3ciLCJSZXNv3M6b2Rwczoq0nByb2plY3RzL3Rlc3Rfd29ya3Nob3AwMDEvaW5zdGFuY2UzLzIwMTkwjI2Z2E2ZTUtaW0iXX1dLCJWZXJzaW9uIjo1MSJ9
Job Queueing...
Summary:

+-----+-----+
| education | num |
+-----+-----+
```

 **Note** The Logview page of each job is valid for seven days.

UI elements

This section describes elements on the Logview UI.



The Logview page consists of two sections:

- ODPS Instance
- ODPS Tasks

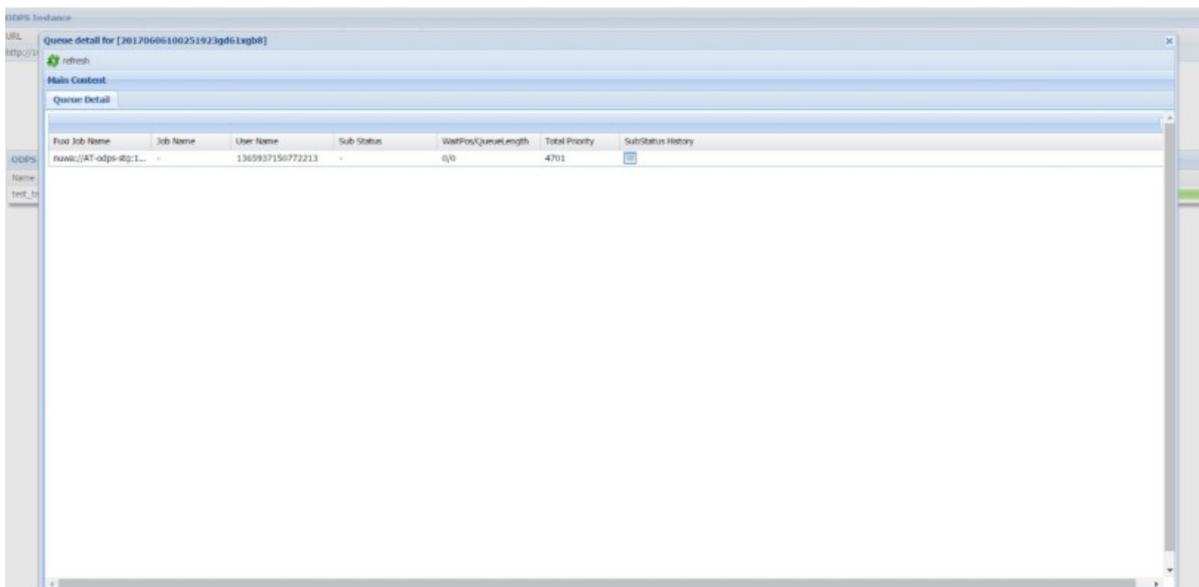
The ODPS Instance section shows the information of the instance that corresponds to the submitted SQL job. The information includes URL, Project, InstanceID, Owner, StartTime, EndTime, and Status.

- You can click the value in the Status column to view queue information. Valid values of Status:
 - Waiting: The job is being processed in MaxCompute and has not been submitted to Job Scheduler.
 - Waiting List: n: The job has been submitted to Job Scheduler and is queued. n indicates the order number of the job in the queue.

- Running: The job is running in Job Scheduler.

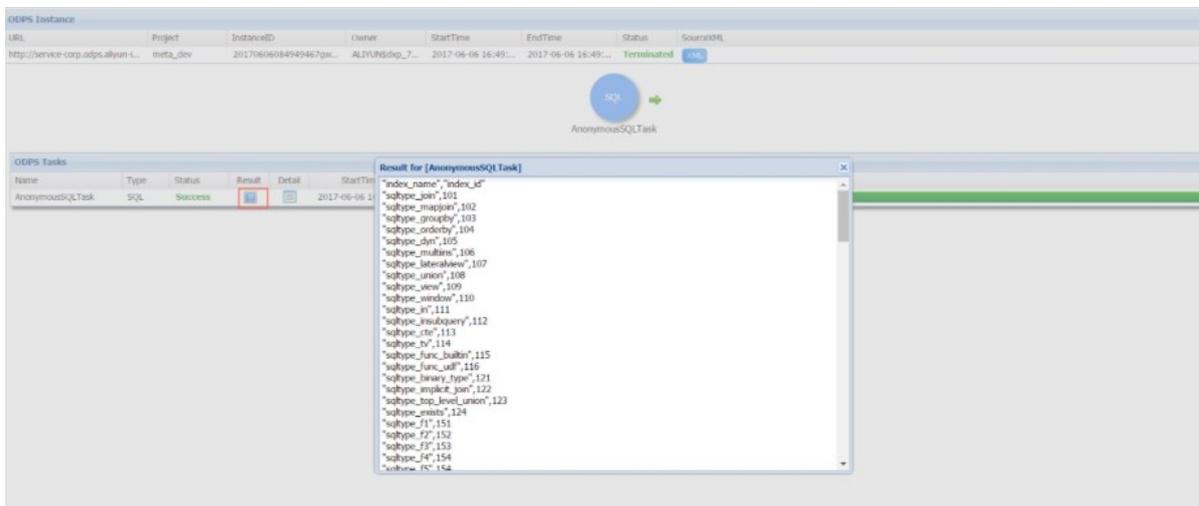
Note If the value of Status is Terminated, the job is complete and has no queue information.

- After you click the value of Status, the following queue information is displayed:
 - Sub Status: the current sub-status information.
 - WaitPos: the position of the job in the queue. The value 0 indicates that the job is running. The value - indicates that the job has not been submitted to Job Scheduler.
 - QueueLength: the total queue length in Job Scheduler.
 - Total Priority: the running priority assigned by the system.
 - SubStatus History: You can click the icon in this column to view the status history. This includes the status code, status description, start time, and duration of a state. The information is unavailable in some versions.

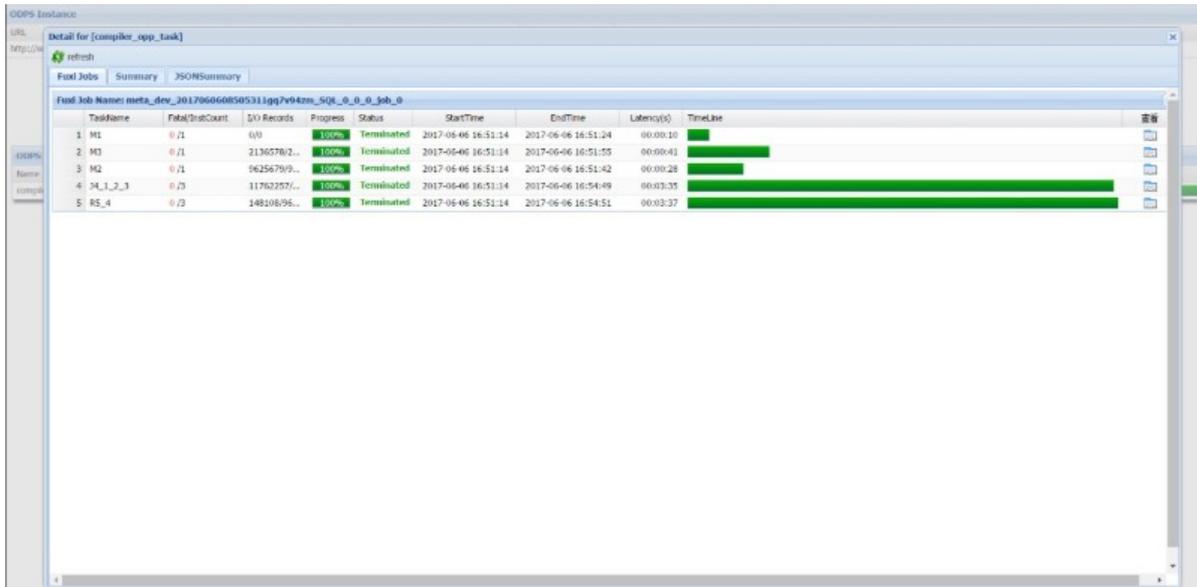


The ODPS Tasks section shows the information of the task that corresponds to the instance. The information includes Name, Type, Status, Result, Detail, Start Time, End Time, Latency (s), and Timeline. Like on other pages, Latency (s) indicates the running duration.

- Result: After a job is complete, you can click the icon in the Result column to view execution results. The following figure shows the execution results of a SELECT statement.



- Detail: You can click the icon in the **Detail** column to view running details. You can view the details of both running and completed jobs.



- In the dialog box that shows details about a MaxCompute job, you can view the following information:
 - A MaxCompute job consists of one or more Fuxi jobs. For example, if a complex SQL job is submitted, MaxCompute automatically submits multiple Fuxi jobs to Job Scheduler.
 - Each Fuxi job consists of one or more Fuxi tasks. For example, a simple MapReduce job generates two Fuxi tasks: map task (M1) and reduce task (R2). If an SQL job is complex, multiple Fuxi tasks may be generated.
 - You can view the name of a Fuxi task. In most cases, a task name consists of letters and digits. A letter represents the type of a task, such as M for a map task. The digits that follow the letter represent the ID and dependencies of a task. For example, R5_4 indicates that the reduce task can be executed only after the J4 task is complete. J4_1_2_3 indicates that the join task can be executed only after the M1, M2, and M3 tasks are complete.
 - I/O Records indicates the numbers of the input and output records of the Fuxi task.
- To view the information of the instance that corresponds to a Fuxi task, you can click the icon in the Show Detail column of the Fuxi task or double-click the Fuxi task.

Note Each Fuxi task consists of one or more Fuxi instances. If the amount of input data for a Fuxi task increases, MaxCompute starts more nodes for the task to process the data. A node is equivalent to a Fuxi instance.

In the lower part of the dialog box, the status information about Fuxi instances at different stages is displayed in groups. For example, you can click the Failed tab to view the nodes where errors occurred. You can click the icon in the StdOut column to view standard output information. You can also click the icon in the StdErr column to view standard error information.

Note To-be-displayed information written in the submitted MaxCompute job is also displayed in the standard output information and standard error information.

Use Logview to handle issues

Tasks with errors: If an error occurs during a task, find the task and click the icon in the Result column in the lower part of the Logview page to view the error information. You can also click the icon in the StdErr column of a Fuxi instance in the details dialog box to view the error information of the instance.

Data skew: The long tail of one or more Fuxi instances may decelerate the execution of a Fuxi task. The long tail is caused by uneven data distribution in a task. After the task is complete, you can view the running results on the Summary tab of the details dialog box. The following output provides an example of the running results.

```
output records:
R2_1_Stg1: 199998999 (min: 22552459, max: 177446540, avg: 99999499)
```

If the difference between the min and max values is large, data skew occurs. For example, if a specific value appears more often than other values in a column, data skew occurs when you execute a JOIN operation based on this column.

1.4.13. Use Logview V2.0 to view job running information

This topic describes the entry and features of Logview V2.0. You can use Logview V2.0 to view job running information.

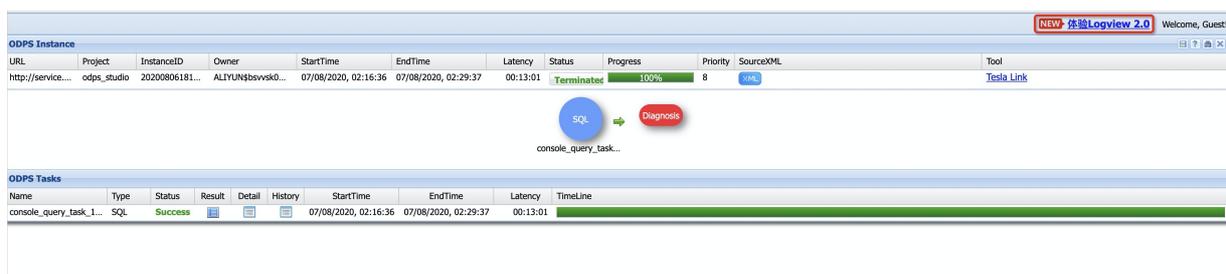
Overview

Logview V2.0 provides a newly designed UI, increases the loading speed, and delivers the following new features:

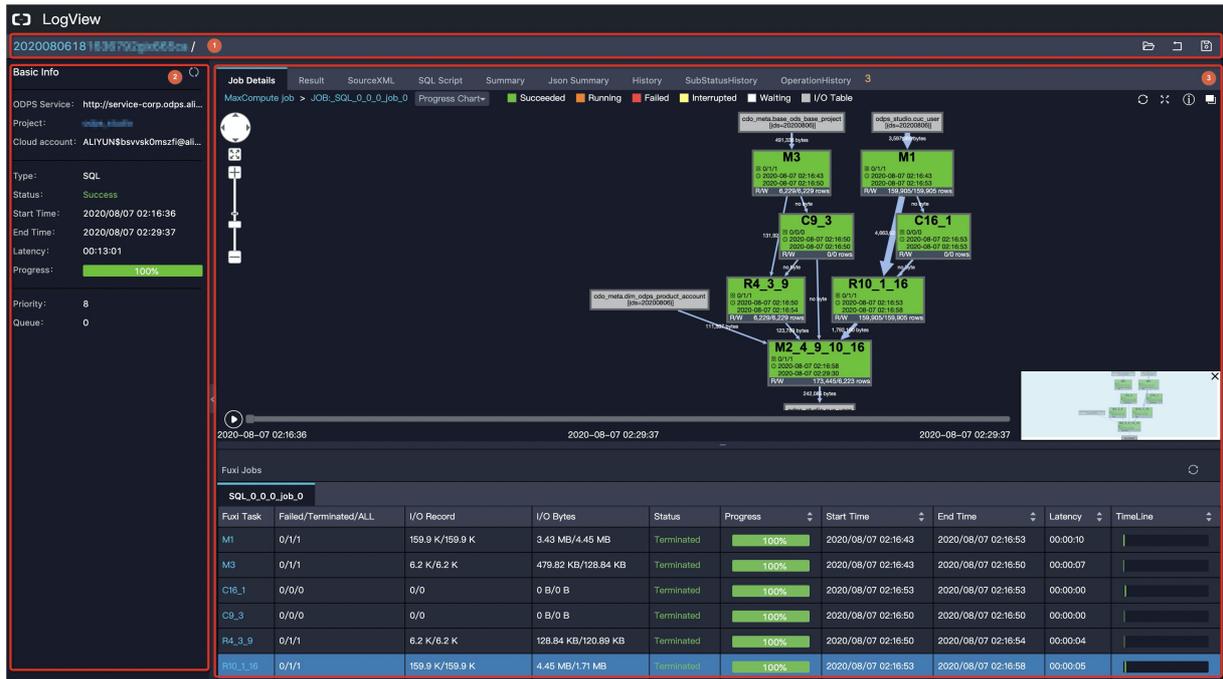
- Provides an interactive directed acyclic graph (DAG) to display the logical architecture of job processing. You can also view the operators of a job.
- Supports job execution playback.
- Allows you to view memory usage and CPU utilization by using Fuxi Sensor.

Entry

After you use the MaxCompute client (odpscmd) to submit a job, the system generates a Logview URL. Enter the Logview URL in the search engine and press Enter. On the Logview UI, click **Go to Logview V2.0**.



Logview V2.0 page



No.	Section
①	The title and function section.
②	The Basic Info section.
③	The job details section.

Title and function section

This section shows the job ID and job name. The job ID uniquely identifies a MaxCompute job. The job ID is generated when you submit the job. The job name is displayed only if the job is submitted by using SDKs. You can also click the icons on the right of this section to perform operations.

Icon	Description
	Open the Logview_detail.txt file that contains job details. The file is saved on your computer.
	Return to the original Logview UI.
	Save the job details as a file to your computer.

Basic Info

This section shows the basic information about a job.

Parameter	Description
MaxCompute Service	The endpoint of MaxCompute on which the job is running.

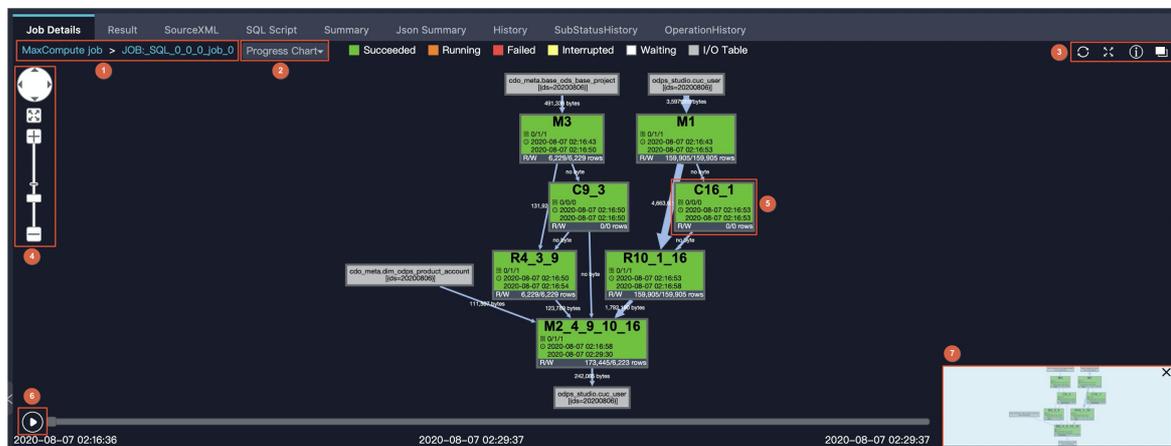
Parameter	Description
Project	The name of the MaxCompute project to which the job belongs.
Cloud account	The Alibaba Cloud account that is used to submit the job.
Type	The type of the job. Valid values: SQL, SQLRT, LOT, XLib, CUPID, AlgoTask, and Graph.
Status	The status of the job. Valid values: <ul style="list-style-type: none"> Success: The job succeeds. Failed: The job fails. Canceled: The job is canceled. Waiting: The job is being processed in MaxCompute but is not submitted to Job Scheduler. Running: The job is being processed in Job Scheduler. Terminated: The job is complete.
Start Time	The time when the job is submitted.
End Time	The time when the job is complete.
Latency	The period during which the job is executed.
Progress	The progress of the job.
Priority	The priority of the job.
Queue	The position of the job in the queue in the resource quota group.

Job details

In the job details section, you can query details about a job. This section consists of the following tabs:

- Job Details
 - Progress chart

In the upper part of the **Job Details** tab, the progress chart of a job is displayed. The progress chart shows the subtask dependencies from three dimensions: Fuxi jobs, Fuxi tasks, and operators. It also provides a series of tools to help you troubleshoot issues. The following figure shows the upper part of the job details tab.



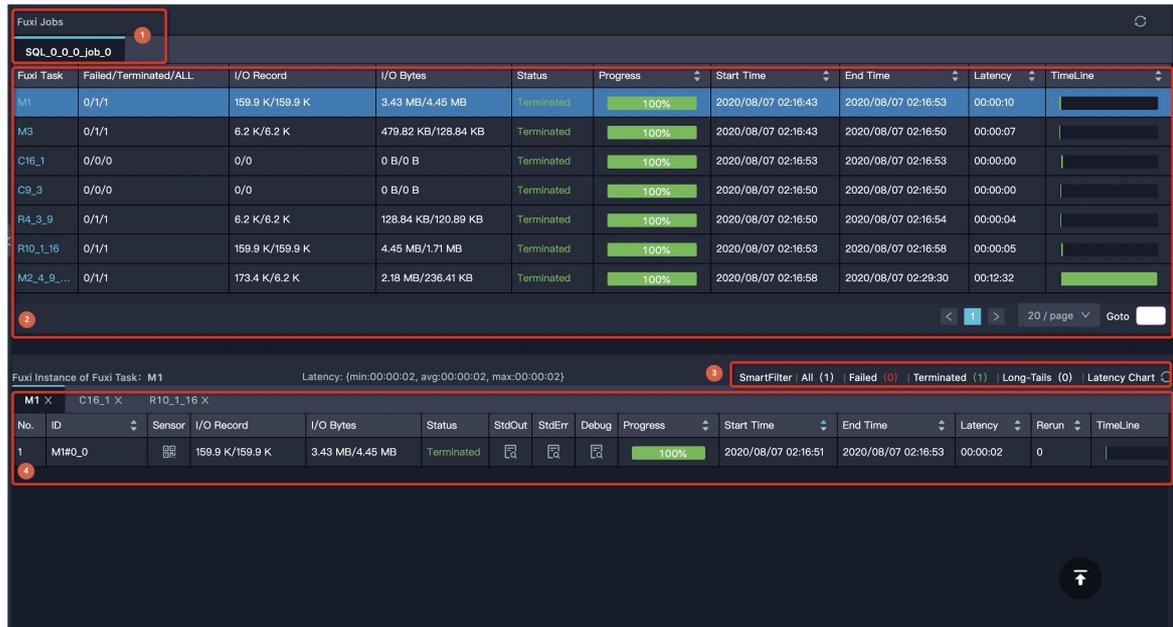
No.	Description
1	The breadcrumb navigation that is used to switch Fuxi jobs. JOB:_SQL_0_0_0_job_0 is the name of a Fuxi job.
2	The troubleshooting tool. You can use Progress Chart, Input Heat Chart, Output Heat Chart, TaskTime Heart Chart, and InstanceTime Heart Chart for troubleshooting.
3	You can click the  icon to refresh the job status and the  icon to zoom in or out the progress chart. You can also click the  icon to obtain MaxCompute Studio documentation and the  icon to switch to the upper level of the job.
4	The zoom tool.
5	<p>The Fuxi task. A MaxCompute job consists of one or more Fuxi jobs. Each Fuxi job consists of one or more Fuxi tasks. Each Fuxi task consists of one or more Fuxi instances. If the amount of input data increases, MaxCompute starts more nodes for each task to process the data. A node is equivalent to a Fuxi instance. For example, a simple MapReduce job generates two Fuxi tasks: map task (M1) and reduce task (R2). If an SQL statement is complex, multiple Fuxi tasks may be generated.</p> <p>You can view the name of each Fuxi task on the execution page. For example, M1 indicates a map task. The 3 and 9 fields in R4_3_9 indicate that the map task can be executed only after M3 and C9_3 are complete. Similarly, M2_4_9_10_16 indicates that the M2 task can be executed only after R4_3_9, C9_3, R10_1_16, and C16_1 are complete. R/W indicates the numbers of rows that the task reads and writes.</p> <p>Click or right-click a task to view operator dependencies and operator graphs of the task.</p> <p>Logview V2.0 provides table dependencies so that you can quickly view the input and output tables.</p>
6	The Fuxi task playback. You can click the  icon to start or stop the playback. You can drag the progress bar to play at a specific time. The start time and end time are displayed on the sides of the progress bar. The playing time is displayed in the middle.
7	EagleEye.

 **Note**

- The playback feature is not applicable to Fuxi tasks that are in the Running state.
- An AlgoTask job, such as a Machine Learning Platform for AI (PAI) job, contains only one Fuxi task. Therefore, no progress charts are provided for these jobs.
- For non-SQL jobs, only Fuxi jobs and Fuxi tasks are displayed.
- If only one Fuxi job exists, the progress chart shows the dependencies among Fuxi tasks. If multiple Fuxi jobs exist, the progress chart shows the dependencies among the Fuxi jobs.

o Running status of jobs

In the lower part of the **Job Details** tab, the detailed running information about the job is displayed. The following figure shows the lower part of the Job Details tab.

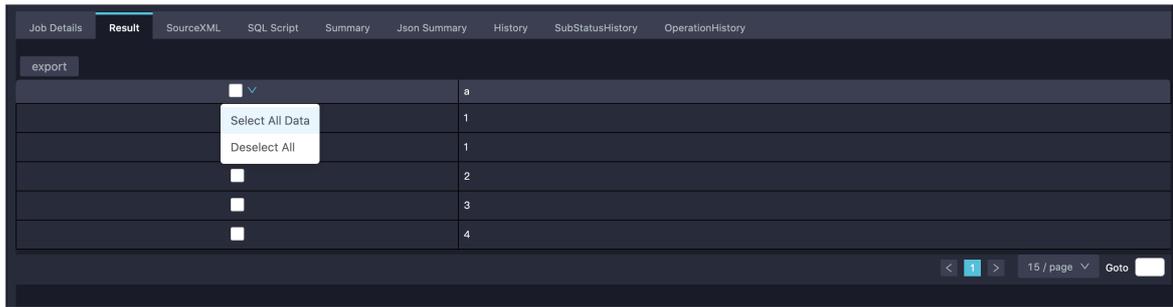


No.	Description
1	The Fuxi Jobs tab. You can switch Fuxi jobs on this tab.
2	The details about Fuxi tasks of the Fuxi job. Click a Fuxi task to display information about the Fuxi instance of this task. By default, information about the Fuxi instance of the first Fuxi task for the first Fuxi job is displayed.
3	Logview divides instances into groups based on their status. You can click the number next to Failed to query information about faulty nodes.
4	Fuxi Sensor. This feature is provided only for AlgoTask jobs, such as PAI jobs. You can use Fuxi Sensor to view the memory usage and CPU utilization of Fuxi instances.
5	StdOut and StdErr. You can view output messages, error messages, and information to be displayed. You can also download the information.
6	Debug. You can debug and troubleshoot errors.

• Result

This tab shows the result of a job. If a job failed, this tab is displayed and shows the cause of the failure. The system displays data in plain text or a table based on the response format specified in MaxCompute. You can log on to the MaxCompute client and set `odps.sql.select.output.format` to specify the format. If you set this parameter to `HumanReadable`, the result is displayed in plain text. Otherwise, the result is displayed in the CSV format.

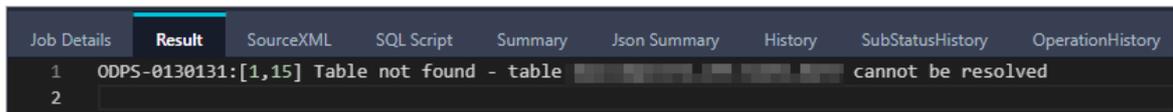
o Table



You can perform the following operations on this tab:

- Select a row to export its data.
- Select the table heading to export data from the current page.
- In the table heading, click the icon and select **Select All Data** or **Deselect All** as needed.
- Click **export** to export data in the CSV format. CSV is a file name extension. You can use a text file or Sublime Text to open CSV files.

o Plain text



• SourceXML

This tab shows the source XML information about the job that is submitted to MaxCompute.

• SQL Script

This tab shows the SQL scripts of the job that is submitted to MaxCompute.

• Summary

This tab shows the overall running information about the job that is submitted to MaxCompute.

• Json Summary

This tab shows the overall running information about the job in the JSON format.

• History

This tab shows the historical information about a Fuxi instance if the instance is executed multiple times.

• SubStatusHistory

This tab shows the detailed information about a job, including the status code, status description, start time, duration, and end time.

Fuxi Sensor

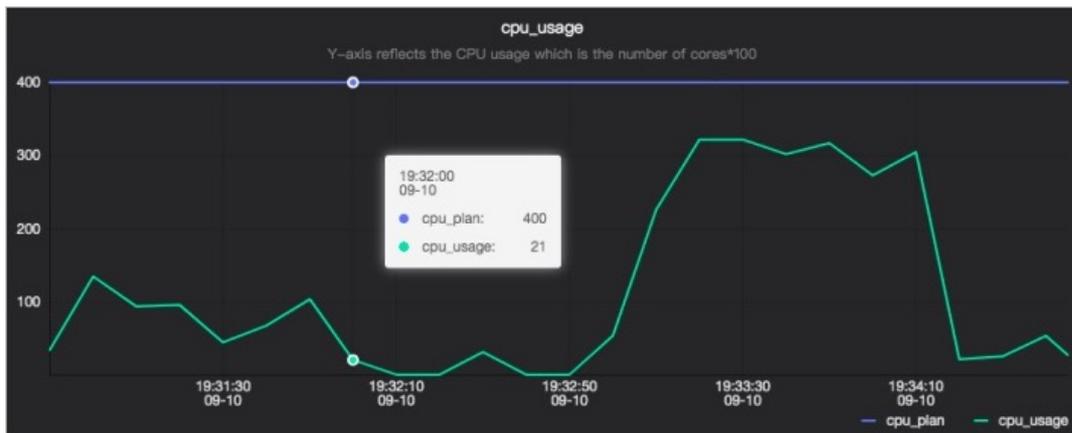
Fuxi Sensor is a resource view that shows a MaxCompute job from all dimensions. You can use Fuxi Sensor to view the memory usage and CPU utilization of a Fuxi instance. You can also use Fuxi Sensor to locate job issues and analyze running performance. For example, you can use Fuxi Sensor in the following scenarios:

- If out-of-memory (OOM) occurs, analyze the amount of memory used.
- Compare the numbers of requested and used resources to optimize the resource request process.

For example, you can use Fuxi Sensor to view the resource usage of a Fuxi instance.

- CPU utilization

The `cpu_usage` chart has two lines. One indicates the number of requested CPUs (`cpu_plan`), and the other one indicates the number of used CPUs (`cpu_usage`). In the y-axis, 400 indicates four processors.

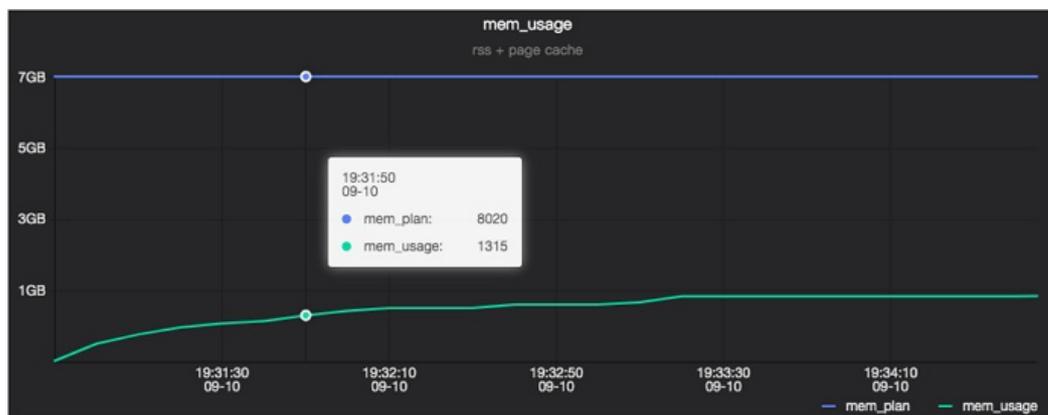


- Memory usage

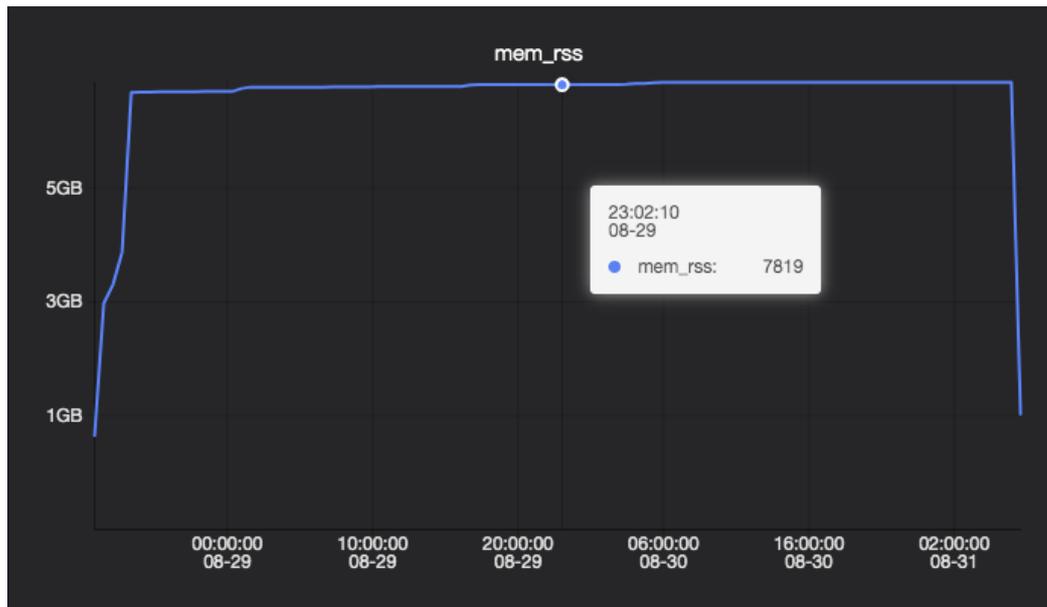
The `mem_usage` chart has two lines. One indicates the number of requested memory resources (`mem_plan`), and the other one indicates the number of used memory resources (`mem_usage`).

`mem_usage` contains Resident Set Size (RSS) and PageCache. RSS indicates the memory that is allocated after kernel page faults occur. This applies when you call Malloc to request memory by using non-file mappings. If the memory is insufficient, RSS cannot be reclaimed. PageCache is the memory occupied by the kernel to cache files that are required by the read and write requests, such as log files. If the memory is insufficient, PageCache can be reclaimed.

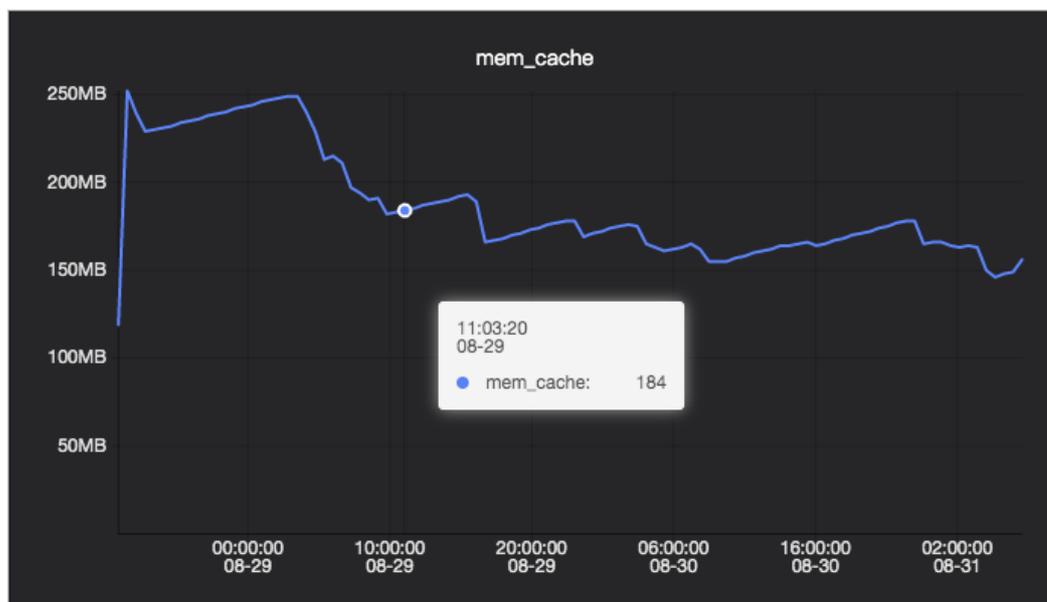
- Memory details



- RSS usage



- PageCache usage



1.5. Basic concepts and common commands

1.5.1. Terms

This topic introduces the basic terms of MaxCompute.

project

A basic organizational unit of MaxCompute. Like a database or schema in a traditional database system, a project is the basic unit of multi-user isolation and access control. A user can have permissions on multiple projects. After authorization, you can access objects such as tables, resources, functions, and instances in a project from another project.

You can run the use project command to access a project. Example:

```
use al_project
-- Enter a project named al_project.
```

Note After you run the preceding command, you are navigated to a project named al_project and manage objects that belong to this project, regardless of which project you are in.

table

A data storage unit of MaxCompute. Logically, a table is a two-dimensional structure that consists of rows and columns, in which each row represents a record and each column represents the values of the same data type. One record can contain one or more columns. The column names and types constitute the schema of this table. All data in MaxCompute is stored in tables. Table columns support all data types in MaxCompute. Tables are the input and output objects of all MaxCompute computing tasks. You can create a table, delete a table, and import data to a table.

You can define partitions for a table to process data more efficiently. You can specify some fields in the table as partition key columns. Partitions within a table are similar to directories within a file system. Each value of a partition key column is called a partition in MaxCompute. You can group multiple fields of a table to a single partition to create a multi-level partition. Multi-level partitions are similar to multi-level directories. If you specify the name of the partition that you want to access, MaxCompute scans only the specified partition. This makes data processing more efficient and reduces cost.

data type

The values in a column of a MaxCompute table must be of a specific data type. MaxCompute supports the following data types.

Basic data types

Type	New in MaxCompute V2.0	Constant	Description	Valid value
TINYINT	Yes	1Y,-127Y	The 8-bit signed integer type.	-128 to 127
SMALLINT	Yes	32767S,-100S	The 16-bit signed integer type.	-32768 to 32767
INT	Yes	1000,-15645787	The 32-bit signed integer type.	-2^{31} to $2^{31}-1$
BIGINT	No	100000000000L,-1L	The 64-bit signed integer type.	$-2^{63}+1$ to $2^{63}-1$
STRING	No	"abc",'bcd',"alibaba", 'inc'	The string, which supports UTF-8 encoding. The character behavior encoded in other formats is not defined.	The size of all values in a column of the STRING type cannot exceed 8 MB.
FLOAT	Yes	None	The 32-bit binary floating point type.	N/A
BOOLEAN	No	True,False	The Boolean type.	True and False

Type	New in MaxCompute V2.0	Constant	Description	Valid value
DOUBLE	No	3.1415926 1E+7	The 64-bit binary floating point type.	-1.0 10^{308} to 1.0 10^{308}
DATETIME	No	Datetime '2017-11-11 00:00:00'	The date and time type. The standard system time is UTC+8.	0001-01-01 00:00:00 000 to 9999-12-31 23:59:59 999
DECIMAL	No	3.5BD, 99999999999.99999 99BD	The precise numeric type based on the decimal system.	Integer: $-10^{36} + 1$ to $10^{36} - 1$ Fractional: round to 10^{-18}
VARCHAR	Yes	None	The variable-length type. n specifies the length.	1 to 65535
BINARY	Yes	None	The binary data type.	The size of all values in a column of the BINARY type cannot exceed 8 MB.
TIMESTAMP	Yes	Timestamp '2017-11-11 00:00:00.123456789'	The timestamp type. The timestamp is independent of time zones.	0001-01-01 00:00:00 000000000 to 9999-12-31 23:59:59 999999999

If you want to use the new data types in MaxCompute V2.0, you must first add the following flags to enable the new data types: `set odps.sql.type.system.odps2=true;` (at the session level) or `setproject odps.sql.type.system.odps2=true;` (at the project level). Otherwise, the following error may occur: `xxxx type is not enabled in current mode`. The data types listed in the preceding table can contain NULL values.

 **Note** Only lowercase letters can be used in the preceding flags.

Take note of the following points when you use the new data types in MaxCompute V2.0:

- The `set odps.sql.type.system.odps2=true;` statement has the following major impacts:
 - The semantics of the INT keyword changes. INT in an SQL statement indicates a 32-bit integer after new data types are enabled.

- The semantics of an integer constant changes. `Select 1 + a;` is used in this example.
 - If the new data types are disabled, the integer constant is processed as the BIGINT type. If the length of the constant exceeds the range for the BIGINT type, the integer constant is processed as the DOUBLE type.
 - If the new data types are enabled, the integer constant is processed as the INT type. In this example, the constant 1 is a 32-bit integer.
 - If the integer constant is processed as the INT type, inconsistency may occur in function prototypes during subsequent operations. In addition, the actions of peripheral tools and subsequent operations might be changed due to the tables that contain new data types and that are generated after data is written to a disk.

- The rules for implicit conversions of data types change.

If the new data types are enabled, some implicit type conversions may be disabled. For example, if the data type is converted from STRING to BIGINT, from STRING to DATETIME, from DOUBLE to BIGINT, from DECIMAL to DOUBLE, or from DECIMAL to BIGINT, precision may be reduced, or errors may occur. In this case, you can use the CAST function to force the data type conversion.

Implicit type conversions greatly affect functions and INSERT statements. For example, an INSERT statement can be executed if the new data types are disabled, but returns an error if the types are enabled.

- If the new data types are disabled, some operations and built-in functions that use data of new data types as parameters and return values are ignored. These operations and functions can be used after the new data types are enabled.
 - Some built-in functions can be used only after the new data types are enabled. The built-in functions include the functions that return the values of the INT type, such as YEAR, QUARTER, MONTH, DAY, HOUR, MINUTE, SECOND, MILLISECOND, NANOSECOND, DAYOFMONTH, and WEEKOFYEAR. The parameters of these functions can be of the INT type, and the BIGINT overload is used for these functions. These functions can be implemented by using the DATEPART built-in function.
 - The resolution of user-defined functions (UDFs) changes.
- The parsing of the BIGINT keyword changes.
- The data types supported for partition key columns change.
 - If the new data types are disabled, the data type of a partition key column can only be STRING.
 - If the new data types are enabled, the data type of a partition key column can be STRING, VARCHAR, CHAR, TINYINT, SMALLINT, INT, or BIGINT.
 - If the new data types are disabled, partition fields in INSERT operations are processed as the STRING type.

- The execution rules of LIMIT statements in SET operations change.

The `Select * from t1 union all select * from t2 limit 10;` statement is used in this example.

- If the new data types are disabled, the preceding statement is expressed as `Select * from t1 union all select * from (select * from t2 limit 10) t2;`
- If the new data types are enabled, the preceding statement is expressed as `Select * from (select * from t1 union all select * from t2) limit 10;`

These changes also apply to UNION, INTERSECT, EXCEPT, ORDER BY, DISTRIBUTE BY, SORT BY, and CLUSTER BY.

- The parsing of data types in IN expressions changes.
 - The `a in (1, 2, 3)` expression is used in this example.
 - If the new data types are disabled, all the values in the parentheses () must be of the same type.
 - If the new data types are enabled, all the values in the parentheses () are implicitly converted to the same type.
- If a constant of the INT type exceeds the range for the INT type, the constant is processed as the BIGINT type. If the constant exceeds the range for the BIGINT type, the constant is processed as the DOUBLE type. If `odps.sql.type.system.odps2` is not set to true, MaxCompute retains the conversion and notifies you that the INT data is processed as the BIGINT type. If `odps.sql.type.system.odps2` is not set to true in your scripts, we recommend that you rewrite your scripts and convert these types into BIGINT to prevent confusion.
- VARCHAR constants can be implicitly converted into STRING constants.
- STRING constants can be combined. For example, `'abc'` and `'xyz'` can be combined as `'abcxyz'`. Different parts can be written in different rows.
- Time values do not include the millisecond component. You can add `-dfp` to Tunnel commands to display milliseconds in the time values.

MaxCompute supports complex data types. The following table lists their definitions and constructors.

Complex data types

Type	Definition	Constructor
ARRAY	<pre>array< int >; array< struct< a:int, b:string >></pre>	<pre>array(1, 2, 3); array(array(1, 2); array(3, 4))</pre>
MAP	<pre>map< string, string >; map< smallint, array< string >></pre>	<pre>map("k1", "v1", "k2", "v2"); map(1S, array('a', 'b'), 2S, array('x', 'y'))</pre>
STRUCT	<pre>struct< x:int, y:int>; struct< field1:bigint, field2:array< int>, field3:map< int, int>></pre>	<pre>named_struct('x', 1, 'y', 2); named_struct('field1', 100L, 'field2', array(1, 2), 'field3', map(1, 100, 2, 200))</pre>

If PyODPS is used, you can use one of the following methods to enable the new data types:

- Execute `o.execute_sql('set odps.sql.type.system.odps2=true;query_sql', hints={"odps.sql.submit.mode": "script"})` to enable the new data types.
- Use DataFrame to enable the new data types. For you can set the hints parameter to specify an immediately executed method, such as `persist`, `execute`, or `to_pandas`. The settings in the following figure are valid only for a single job.

```
from odps.df import DataFrame
users = DataFrame(o.get_table('odps2_test'))
users.persist('copy_test', hints={'odps.sql.type.system.odps2': 'true'})
```

- If you use DataFrame to enable the new data types and want the settings to take effect globally, configure `options.sql.use_odps2_extension = True`.

resource

A unique concept in MaxCompute. To implement UDF and MapReduce features, you must use resources.

- MaxCompute SQL UDFs: After you write a UDF, you must package it into a JAR file and upload the package to MaxCompute as a resource. When you run the UDF, MaxCompute automatically downloads the JAR file and obtains the code to run the UDF. JAR files are a type of MaxCompute resources. When you upload a JAR file, a resource is created in MaxCompute.
- MaxCompute MapReduce: After you write a MapReduce program, you must package it into a JAR file and upload the file to MaxCompute as a resource. When you run a MapReduce job, the MapReduce framework automatically downloads the JAR file and obtains the code to run the MapReduce job.

Note

- Some limits are imposed on how MaxCompute UDFs and MapReduce access resources. For more information, see [Limits](#).
- You can also upload tables or text files to MaxCompute as different types of resources. You can read or use these resources when you run UDFs or MapReduce jobs. MaxCompute provides APIs for you to read and use resources. For more information, see [UDTF description](#).

MaxCompute supports the following resource types:

- File
- Table: tables in MaxCompute.
- JAR: compiled JAR files.
- Archive: compressed files identified by the resource name extension. Supported file types include .zip, .tgz, .tar, and .jar.
- Py: Python scripts used by Python UDFs.

 Note For more information about resource operations, see [Resource operations](#).

UDF

MaxCompute provides SQL computing capabilities. You can use the built-in functions in MaxCompute SQL statements to complete some computing and counting tasks. If these built-in functions do not meet your requirements, you can use the Java APIs that MaxCompute provides to develop UDFs. UDFs can be further divided into user-defined scalar-valued functions, user-defined aggregate functions (UDAFs), and user-defined table-valued functions (UDTFs).

After you write the UDF code, you must package the code into a JAR file, upload it to MaxCompute as a resource, and then register this UDF in MaxCompute. To use a UDF in MaxCompute, you need only to specify its name and parameters in an SQL statement as you do when you use the built-in functions of MaxCompute.

 Note For more information about UDF operations, see [Function operations](#).

task

A basic computing unit of MaxCompute. Both SQL and MapReduce features are implemented as tasks. MaxCompute parses most of the tasks that you submit, especially computing tasks, such as SQL DML statements and MapReduce tasks. Then, MaxCompute generates a task execution plan based on the parsing results.

An execution plan consists of several dependent stages. An execution plan can be logically defined as a directed graph. Vertices of the graph represent stages, and edges of the graph represent dependencies between stages. MaxCompute executes the stages based on the dependencies in the graph (execution plan). A stage has multiple processes, also known as workers. The workers in each stage cooperate to complete computations for the stage. Different workers in a stage process different data, but they all use the same execution logic.

When you run a computing task, the task is converted into an instance. You can perform operations on this instance, such as obtaining status information and terminating the instance.

Some MaxCompute tasks, such as SQL DDL statements, are not computing tasks. These tasks only read and modify metadata in MaxCompute. MaxCompute cannot generate execution plans for these tasks.

 **Notice** MaxCompute does not convert all requests into tasks. For example, project, resource, UDF, and instance operations are not executed as tasks.

instance

Some MaxCompute tasks are converted into instances during the execution process. An instance has two stages: Running and Terminated. Instances that are in the Running stage are also in the Running state. Instances that are in the Terminated stage can be in the Success, Failed, or Canceled state. You can query or modify the status of a running task by using the instance ID provided by MaxCompute. Example:

```
status <instance_id>;  
-- Query the status of an instance.  
kill <instance_id>;  
-- Terminate an instance. After you run the kill command, the status of the instance changes to Canceled.
```

resource quota

A per-process limit on the use of system resources. Two types of quotas are involved: storage and computing. The storage quota is the upper limit of the storage configured for a project. If the storage usage approaches the storage quota, an alert is triggered. The computing quota limits the use of memory and CPU resources. The memory and CPUs for the processes that are concurrently run in a project cannot exceed the computing quota.

1.5.2. Common commands

1.5.2.1. Overview

MaxCompute allows you to perform operations on objects, such as projects, tables, resources, and instances. You can run command and execute statements on the client or use SDKs to perform operations on these objects.

This topic describes how to run the commands and execute the statements on the MaxCompute client.

-  **Note**
- For more information about how to download and configure the client, see [Configure the client](#).
 - For more information about the SDKs, see [SDK for Java](#).

1.5.2.2. Project operations

This topic describes common statements for project operations.

Create or delete a project

MaxCompute does not provide statements for you to create or delete a project. You can create or delete a project in the Apsara Uni-manager Management Console.

- For more information about how to create a project, see [Create a project](#).
- If you want to delete a project, you can log on to the Apsara Uni-manager Management Console and perform the following steps: Choose **Products > Big Data > MaxCompute**. In the left-side navigation pane, click **Projects**. Find the project that you want to delete and click **Delete** in the Actions column.

Go to a project

Syntax:

```
use <project_name>;
```

Description: This statement is used to go to a specified project. After you go to the project, you can directly manage all objects in the project.

 **Note** If the specified project does not exist or you are not added to the project, the system returns an error and exits.

Example:

```
use my_project;
-- my_project is the project that you have permissions to access.
```

 **Note**

- The preceding statement is executed in the MaxCompute client.
- All the statement keywords, project names, table names, and column names in MaxCompute are not case-sensitive.

If the test_src table exists in the my_project project, execute the following statement:

```
select * from test_src;
```

MaxCompute automatically searches for the tables from the my_project project. If the table exists, its data is returned. Otherwise, the system returns an error and exits.

If you are in the my_project project and want to access the test_src table in the my_project2 project, you must specify the project name. Execute the following statement to access the test_src table in the my_project2 project:

```
select * from my_project2.test_src;
```

Data of the my_project2 project, instead of data of the test_src table in the my_project project, is returned.

View projects

Syntax:

```
list projects;
```

Description: This statement is used to query all projects in MaxCompute.

Clear a project

Syntax of the statement that is used to view objects in the project recycle bin:

```
show recyclebin;
```

Description: This statement is used to query all objects in the project recycle bin.

 **Note** Only the project owner is allowed to execute this statement.

Syntax of the statement that is used to clear the project recycle bin:

```
purge all;
```

Description: This statement is used to clear the project recycle bin to release storage space.

 **Note** Only the project owner is allowed to execute this statement.

Syntax of the statement that is used to clear a table:

```
purge table tblname;
```

Description: This statement is used to clear a specified table from the recycle bin to release storage space.

 **Note**

- If the specified table exists, the project owner and users who have write permissions on the table are allowed to execute this statement.
- If the table has been deleted, only the project owner is allowed to execute this statement.

1.5.2.3. Table operations

This topic describes common statements for table operations.

CREATE TABLE

Syntax:

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table_name
[(col_name data_type [DEFAULT value] [COMMENT col_comment], ...)]
[COMMENT table_comment]
[PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]
[LIFECYCLE days];
CREATE TABLE [IF NOT EXISTS] table_name
[AS select_statement | LIKE existing_table_name];
```

Description: This statement is used to create a table.

Examples:

```
CREATE TABLE IF NOT EXISTS sale_detail( shop_name STRING,
customer_id STRING, total_price DOUBLE)
PARTITIONED BY (sale_date STRING,region STRING);
-- Create a partitioned table named sale_detail if no table with this name exists.
```

Note

- Table and column names are not case-sensitive.
- A table or column name can contain only letters, digits, and underscores (_) and must start with a letter. It can be up to 128 bytes in length. Otherwise, an error is returned.
- A comment must be a valid string and can be up to 1,024 bytes in length. Otherwise, an error is returned.
- For more information about the statement, see [Create a table](#).

CHANGEOWNER

Syntax:

```
ALTER TABLE table_name CHANGEOWNER to new_owner;
```

Description: This statement is used to change the owner of a table.

Examples:

```
ALTER TABLE test1 CHANGEOWNER to 'ALIYUN$xxx@aliyun.com';
-- Change the owner of the test1 table to ALIYUN$xxx@aliyun.com.
```

DROP TABLE

Syntax:

```
DROP TABLE [IF EXISTS] table_name;
```

Description: This statement is used to delete a table.

 **Note** If you do not specify IF EXISTS and the table does not exist, an error is returned. If you specify IF EXISTS, a success message is returned, regardless of whether the table exists.

Parameters:

table_name: the name of the table you want to delete.

Examples:

```
DROP TABLE sale_detail;
-- If the table exists, a success message is returned.
DROP TABLE IF EXISTS sale_detail;
-- A success message is returned regardless of whether the sale_detail table exists.
```

Query table information

Syntax:

```
DESC <table_name>;  
-- Query information about a table or view.  
DESC extended <table_name>;  
-- Query information about an external table.
```

Description: The statements are used to query information about specific tables. Output fields include Owner, Project, CreateTime, LastDDLTime, LastModifiedTime, InternalTable, Size, Native Columns, Partition Columns, and Extended Info. The Owner field indicates the owner of the table. The Project field indicates the project to which the table belongs. The CreateTime field indicates the time when the table was created. The LastDDLTime field indicates the last time when a DDL statement is executed for the table. The LastModifiedTime field indicates the last time when the table data was modified. The InternalTable field indicates that the queried object is a table. The value of this field is always YES. The Size field indicates the storage occupied by table data, in bytes. The Native Columns field shows the names, data types, and comments of standard columns. The Partition Columns field shows the names, data types, and comments of partition key columns. The Extended Info field shows information about an external table, including the storage handler and location.

Parameters:

table_name: In the first statement, this parameter specifies the name of the table or view. In the second statement, this parameter specifies the name of the external table.

Examples:

Execute the following statement to query information about a partitioned table:

```
DESC sale_detail;
```

The following result is returned:

```
+-----+  
| Owner: ALIYUN$odpsuser@aliyun.com | Project: test_project |  
| TableComment: |  
+-----+  
| CreateTime: 2014-01-01 17:32:13 |  
| LastDDLTime: 2014-01-01 17:57:38 |  
| LastModifiedTime: 1970-01-01 08:00:00 |  
+-----+  
| InternalTable: YES | Size: 0 |  
+-----+  
| Native Columns: |  
+-----+  
| Field | Type | Comment |  
+-----+  
| shop_name | string | |  
| customer_id | string | |  
| total_price | double | |  
+-----+  
| Partition Columns: |  
+-----+  
| sale_date | string | |  
| region | string | |  
+-----+
```

 Note

- If the queried object is a non-partitioned table, the Partition Columns field is not displayed.
- If the queried object is a view, the InternalTable field is replaced with VirtualView whose value is always YES, and the Size field is replaced with ViewText. The ViewText field defines View, such as select * from src. For more information about views, see [Create a view](#).

Query partition information

Syntax:

```
desc table_name partition(pt_spec);
```

Description: This statement is used to query information about a partition.

Examples:

Execute the following statement to query information about a partition:

```
desc meta.m_security_users partition (ds='20151010');
```

The following result is returned:

```
+-----+
| PartitionSize: 2109112          |
+-----+
| CreateTime:    2015-10-10 08:48:48      |
| LastDDLTime:   2015-10-10 08:48:48      |
| LastModifiedTime: 2015-10-11 01:33:35   |
+-----+
OK
```

SHOW TABLES and SHOW TABLES LIKE

Syntax:

```
SHOW TABLES;
SHOW TABLES like 'chart';
```

Description:

- SHOW TABLES: queries all tables in the current project.
- SHOW TABLES like 'chart': queries the tables whose names match chart in the current project. You can use regular expressions in the statement to filter tables.

Examples:

Execute the following statement to query all tables in the current project:

```
show tables;
```

Execute the following statement to query the tables whose names match ods_brand:

```
show tables like 'ods_brand*';
```

The following result is returned:

```
ALIYUN$odps_user@aliyun.com:ods_brand  
.....
```

Note

- odps_user@aliyun.com: the name of the user who creates the table.
- table_name: the name of the table.

SHOW PARTITIONS

Syntax:

```
SHOW PARTITIONS <table_name>;
```

Description: This statement is used to query all partitions of a table.

Parameters:

table_name: the name of the table that you want to query. If the table does not exist or is a non-partitioned table, an error is returned.

Examples:

Execute the following statement to query all partitions of a table:

```
SHOW PARTITIONS table_name;
```

The following result is returned:

```
partition_col1=col1_value1/partition_col2=col2_value1  
partition_col1=col1_value2/partition_col2=col2_value2
```

Note

- partition_col1 and partition_col2: the partition key columns of the table.
- col1_value1, col2_value1, col1_value2, and col2_value2: the values in the partition key columns.

1.5.2.4. Instance operations

This topic describes common statements for instance operations.

Show Instances/Show P

Syntax:

```
SHOW INSTANCES [FROM startdate TO enddate] [number];  
SHOW P [FROM startdate TO enddate] [number];  
SHOW INSTANCES [-all];  
SHOW P [-all];  
SHOW P -p <project name>;
```

Description: This statement is used to query information about instances that you have created.

Parameters:

- **startdate** and **enddate**: the beginning and end of the time range to query. Information about the instances created within the specified period is returned. The dates must be in the format of yyyy-mm-dd. These parameters are optional. If they are not specified, information about instances that you have created in the last three days is returned.
- **number**: the number of instances to return. Information about the specified number of instances submitted at the time nearest to the current time is returned in chronological order. If this parameter is not specified, information about all instances that meet the requirements is returned.
- **-all**: indicates that information about the instances executed in the current project is returned. By default, a maximum of 50 records are returned. You can execute this statement only when you have the LIST permission on the current project. To return more records, you must use the **-limit** number option. For example, you can use `show p -all -limit 100` to return information about 100 instances executed in the current project.
- **project name**: the name of the project. The account that you use must be a member of the project.

Output fields include **StartTime**, **RunTime**, **Status**, **InstanceID**, and **Query**. The values of **StartTime** and **RunTime** are in seconds. The **Query** field indicates the SQL statement that corresponds to the instance. The following code provides an example of the output:

Execute the following statement to query the information about instances created by the current user:

```
show p;
```

The following result is returned:

```
StartTime      RunTime Status InstanceID      Owner      Query
2015-04-28 13:57:55 1s  Success 20150428xxxxxxxxxxxxxxxx ALIYUN$xxxxx@aliyun-inner.com  select * from t
ab_pack_priv limit 20;
... ..
... ..
```

An instance can be in one of the following states:

- **Running**: The instance is running.
- **Success**: Instance operations are complete.
- **Waiting**: The instance is waiting to run.
- **Failed**: The job failed, but the data in the destination table is not modified.
- **Suspended**: The instance is suspended.
- **Cancelled**: The instance is stopped.

Status Instance

Syntax:

```
status instance_id;
```

Description: This statement is used to query the status of a specific instance. Valid values: Success, Failed, Running, and Cancelled.

 **Note** If the instance is not created by the current user, an error is returned.

Parameters: **instance_id** indicates the ID of the instance whose status you want to query. This parameter uniquely identifies an instance.

Examples:

Execute the following statement to query the status of the instance whose ID is 20131225123xxxxxxxxxxxxxxxx.

```
status 20131225123xxxxxxxxxxxxxxxx;
```

The following result is returned:

```
Success
```

Top Instance

Syntax:

```
top instance;  
top instance -all;
```

 **Note** Only project owners and administrators can execute the statement.

Description:

- `top instance`: queries information about running jobs that are submitted by the current user in the current project. Output fields include InstanceID, Owner, Type, StartTime, Progress, Status, Priority, RuntimeUsage (CPU/MEM), TotalUsage (CPU/MEM), and QueueingInfo (POS/LEN).
- `top instance -all`: queries information about all running jobs in the current project. By default, a maximum of 50 records are returned. To return more records, use the `-limit` number option.

Kill Instance

Syntax:

```
kill <instance_id>;
```

Description: This statement is used to stop an instance. You can stop an instance only in the Running state. Note that this is an abnormal process. The execution result of this statement only means that the system has received the request. It does not mean that the job has been stopped. Therefore, you must use `status` to query the instance status.

Parameters: `instance_id` indicates the ID of the instance. It must be the ID of a running instance. Otherwise, an error is returned.

Examples:

```
kill 20131225123xxxxxxxxxxxxxxxx;  
-- Stop the instance whose ID is 20131225123xxxxxxxxxxxxxxxx.
```

Desc Instance

Syntax:

```
desc instance <instance_id>;
```

Description: This statement is used to query job information based on the instance ID. Output fields include Query, Owner, Startime, Endtime, and Status.

Parameters: instance_id indicates the ID of the instance.

Examples:

-- Execute the following statement to query information about the job that corresponds to the instance whose ID is 20150715xxxxxxxxxxxxxxxx.

```
desc instance 20150715xxxxxxxxxxxxxxxx;
```

The following result is returned:

ID	20150715xxxxxxxxxxxxxxxx
Owner	ALIYUN\$XXXXXX@alibaba-inc.com
StartTime	2015-07-15 18:34:41
EndTime	2015-07-15 18:34:42
Status	Terminated
console_select_query_task_1436956481295	Success
Query	select * from mj_test;

Wait Instance

Syntax:

```
wait instance_id;
```

Description: This statement is used to obtain the operational logs of the job based on the instance ID. The returned information includes a Logview URL. You can log on to Logview to view log details.

 **Note** You can obtain the Logview URL of an instance created for more than three days. However, you cannot use the URL to go to Logview because it expires and is cleared.

Parameters: instance_id indicates the ID of the instance.

Examples:

Execute the following statement to query the operational logs of the job that corresponds to the instance whose ID is 20170925161122379gxxxxxx.

```
wait 20170925161122379gxxxxxx;
```

The following result is returned:

```
ID = 20170925161122379g3xxxxxx
Log view:
http://logview.odps.aliyun.com/logview/?h=http://service.odps.aliyun.com/api&p=aliam&i=20170925161122379g3xxxxxx
dqp&token=XXXXXXvbi6jEifQ==
Job Queueing...
Summary:
resource cost: cpu 0.05 Core * Min, memory 0.05 GB * Min
inputs:
  alian.bank_data: 41187 (588232 bytes)
outputs:
  alian.result_table: 8 (640 bytes)
Job run time: 2.000
Job run mode: service job
Job run engine: execution engine
M1:
  instance count: 1
  run time: 1.000
  instance time:
    min: 1.000, max: 1.000, avg: 1.000
  input records:
    TableScan_REL5213301: 41187 (min: 41187, max: 41187, avg: 41187)
)
  output records:
    StreamLineWrite_REL5213305: 8 (min: 8, max: 8, avg: 8)
R2_1:
  instance count: 1
  run time: 2.000
  instance time:
    min: 2.000, max: 2.000, avg: 2.000
  input records:
    StreamLineRead_REL5213306: 8 (min: 8, max: 8, avg: 8)
  output records:
    TableSink_REL5213309: 8 (min: 8, max: 8, avg: 8)
```

1.5.2.5. Resource operations

This topic describes common statements for resource operations.

Add a resource

Syntax:

```
add file <local_file> [as alias] [comment 'cmt'][-f];
add archive <local_file> [as alias] [comment 'cmt'][-f];
add table <table_name> [partition (spec)] [as alias] [comment 'cmt'][-f];
add jar <local_file.jar> [comment 'cmt'][-f];
add py <local_file.py> [comment 'cmt'][-f];
```

Description: The statements are used to add resources to MaxCompute.

Parameter	Description
file/archive/table/jar/py	The type of the resource.

Parameter	Description
local_file	The path of the file that you want to add. The file name is used as the resource name. A resource name is a unique identifier of a resource.
table_name	The name of the MaxCompute table.
[partition (spec)]	If the resource that you want to add is a partitioned table, MaxCompute takes only a partition, not the whole partitioned table, as a resource.
alias	The name of the resource. If this parameter is not specified, the file name is used as the resource name. This feature does support JAR and PY resources.
[comment 'cmt']	The comment of the resource.
[-f]	If a resource with the same name exists, this operation overwrites the existing resource. If this parameter is not specified and a resource with the same name exists, the operation fails.

Examples:

-- Add a table resource whose alias is sale.res to MaxCompute.

```
add table sale_detail partition (ds='20170602') as sale.res comment 'sale detail on 201706 02' -f;
```

The following result is returned:

```
OK: Resource 'sale.res' have been updated.
```

 **Note** The size of each resource file cannot exceed 500 MB. The total size of resource files referenced by a single SQL or MapReduce job cannot exceed 2,048 MB.

Delete a resource

Syntax:

```
DROP RESOURCE <resource_name>;
```

Description: This statement is used to delete an existing resource.

Parameters:

resource_name: the name of the resource that you want to delete.

Examples:

Execute the following statement to delete an existing resource:

```
DROP RESOURCE getaddr.jar;
```

The following result is returned:

```
Confirm to "DROP RESOURCE getaddr.jar" (yes/no)? y
-- Enter y to confirm your input.
OK
```

Query resources

Syntax:

```
LIST RESOURCES;
```

Description: This statement is used to query all resources in the current project.

Examples:

Execute the following statement to query all resources in the current project:

```
list resources;
```

The following result is returned:

Resource Name	Comment	Last Modified Time	Type
1234.txt	2014-02-27 07:07:56	file	
mapred.jar	2014-02-27 07:07:57	jar	

Download a resource

Syntax:

```
GET RESOURCE <resource_name> <path>;
```

Description: This statement is used to download a resource from MaxCompute to your computer. You can download file, JAR, archive, and PY resources, but not table resources.

Parameters:

- resource_name: the name of the resource that you want to download.
- path: the path to save the resource.

Examples:

Execute the following statement to download a resource from MaxCompute to your computer:

```
get resource odps-udf-examples.jar d:\;
```

The following result is returned:

```
OK
```

1.5.2.6. Function operations

This topic describes common statements for function operations.

Create a function

Syntax:

```
CREATE FUNCTION <function_name> AS <package_to_class> USING<resource_list>;
```

The following table describes the parameters in this statement.

Parameter	Description
function_name	The name of the user-defined function (UDF). This name is used in SQL statements to reference this function.
package_to_class	For a Java UDF, package_to_class indicates a fully qualified class name. It consists of names from the top-level package name all the way to the class name for UDF implementation. For a Python UDF, package_to_class indicates the Python script name and class name. The value must be enclosed in a pair of single quotation marks ('').
resource_list	The list of resources that the UDF uses. The list must include the resource where the UDF code is located. Make sure that the resources are uploaded to MaxCompute before you register the function. If the user code reads resource files by using the distributed cache API, this list must also include all resource files that are read by the UDF. If the resource list contains multiple resources, separate them with commas (,). The resource list must be enclosed in a pair of single quotation marks (''). If you need to specify the project where the resource is located, use the <project_name>/resources/<resource_name> format.

Examples:

- Assume that Java UDF class org.alidata.odps.udf.examples.Lower is in my_lower.jar. Execute the following statement to create the my_lower function:

```
CREATE FUNCTION test_lower AS 'org.alidata.odps.udf.examples.Lower'
USING 'my_lower.jar';
```

- Assume that Python UDF class MyLower is in the pyudf_test.py script of the test_project project. Execute the following statement to create the my_lower function:

```
create function my_lower as 'pyudf_test.MyLower'
using 'pyudf_test.py';
```

- Assume that Java UDF class com.aliyun.odps.examples.udf.UDTFResource is in udtfexample1.jar and the function depends on file resource file_resource.txt, table resource table_resource1, and archive resource test_archive.zip. Execute the following statement to create the test_udtf function:

```
create function test_udtf as 'com.aliyun.odps.examples.udf.UDTFResource'
using 'udtfexample1.jar, file_resource.txt, table_resource1, test_archive.zip';
```

Note

- Similar to resource file names, function names must be unique.
- Only project owners have the permissions to use UDFs to overwrite built-in functions. If you use a UDF that overwrites a built-in function, warning information is included in the Summary parameter after the SQL statement is executed.

Delete a function

Syntax:

```
DROP FUNCTION <function_name>;
```

Parameters:

function_name: the name of the function you want to delete.

Examples:

```
DROP FUNCTION test_lower;
```

Query functions

Syntax:

```
LIST FUNCTIONS;  
-- Query all UDFs in the current project.  
LIST FUNCTIONS -p project_name;  
-- Query all UDFs in a specific project.
```

1.5.2.7. Time zone configuration operations

This topic describes how to use a SET command to configure the time zone for a MaxCompute project.

Time zone configurations

The default time zone of a MaxCompute project is UTC+8. Time-related built-in functions and the fields of the DATETIME, TIMESTAMP, and DATE types are calculated based on UTC+8. You can use one of the following methods to configure the time zone:

- Session level: Run the `set odps.sql.timezone=<timezoneid>;` command along with a computing statement.

Examples:

Run the following command to set the time zone to Asia/Tokyo:

```
set odps.sql.timezone=Asia/Tokyo;
```

Run the following command to query the current time zone:

```
select getdate();
```

The following result is returned:

```
+-----+  
|_c0 |  
+-----+  
| 2018-10-30 23:49:50 |  
+-----+
```

- Project level: Run the `setProject odps.sql.timezone=<timezoneid>;` command as the project owner.

 **Notice** After the time zone of a project is configured, it is used for all time computing, and the data of existing jobs is affected. Therefore, exercise caution when you perform this operation. We recommend that you set time zones only for new projects.

Limits and usage notes

- SQL built-in date functions, user-defined functions (UDFs), user-defined types (UDTs), user-defined joins (UDJs), and SELECT TRANSFORM allow you to obtain the timezone attribute of a project to configure the time zone.
- A time zone must be configured in the format such as Asia/Shanghai, which supports daylight saving time. Do not configure it in the GMT+9 format.
- If the time zone of the SDK differs from that of the project, you must configure the GMT time zone to convert the data type from DATETIME into STRING.
- After you configure the time zone, the output time may be different from the real time after you run the related SQL statements by using DataWorks. For data that is between the years 1900 and 1928, the time difference is 352 seconds. For data that is before the year 1900, the time difference is 9 seconds.
- MaxCompute, SDK for Java, and the related client are updated to ensure that DATETIME data stored in MaxCompute is accurate across multiple time zones. The updated versions of the client and SDK have the -oversea suffix. The update may affect the display of DATETIME data earlier than January 1, 1928 in MaxCompute.
- If the time zone is not UTC+8, we recommend that you update SDK for Java and the client. This ensures that the SQL-based computing results and data transferred by using Tunnel after January 1, 1900 are accurate and consistent. After the update, for the DATETIME data before January 1, 1900, the SQL-based computing results and data transferred by using Tunnel may still have a difference of 343 seconds. For the DATETIME data between January 1, 1900 and January 1, 1928 uploaded before the update, the time in the new version is 352 seconds earlier.
- If you do not update SDK for Java and the client to the version with the -oversea suffix, the SQL-based computing results and data transferred by using Tunnel still have differences. For data before January 1, 1900, the time difference is 9 seconds. For data between January 1, 1900 and January 1, 1928, the time difference is 352 seconds.

 **Note** The modification of time zone configurations for the new client or SDK for Java does not affect the time zone configuration in DataWorks. Therefore, the time zones may vary. You must evaluate the impact on scheduled tasks in DataWorks.

- If you use a third-party client that is connected to MaxCompute by using JDBC, you must configure the time zone on the client to ensure time consistency between the client and the server.
- MaxCompute MapReduce, Machine Learning Platform for AI, and MaxCompute Graph support the time zone configuration.
- MaxCompute Spark supports the time zone configuration.
 - If tasks are submitted to a MaxCompute computing cluster, the time zone of the project can be automatically obtained.
 - If tasks are submitted from spark-shell, spark-sql, or pyspark in yarn-client mode, you must configure the Spark-defaults.conf parameter of the driver and add spark.driver.extraJavaOptions - Duser.timezone=America/Los_Angeles. The timezone parameter indicates the time zone that you want to use.

1.5.2.8. Tunnel operations

This topic describes common commands for Tunnel operations.

Use of Tunnel commands

You can run the `tunnel help;` subcommand on the client to query help information. The following result is returned:

```
Usage: tunnel <subcommand> [options] [args]
Type 'tunnel help <subcommand>' for help on a specific subcommand.
Available subcommands:
  upload (u)
  download (d)
  resume (r)
  show (s)
  purge (p)
  help (h)
tunnel is a command for uploading data to / downloading data from ODPS.
```

Parameters:

- **upload:** uploads data to a MaxCompute table. You can upload files or level-1 directories to only one table or partition each time. For a partitioned table, you must specify the partition to which you want to upload data. For a multi-level partitioned table, you must specify the last-level partition.

```
tunnel upload log.txt test_project.test_table/p1="b1",p2="b2";
-- Upload data in the log.txt file to the p1="b1" and p2="b2" partitions of the test_table table that has two levels of partitions in the test_project project.
tunnel upload log.txt test_table --scan=true;
-- Upload data in the log.txt file to the test_table table. The scan parameter is used to check whether data in the log.txt file complies with the schema of the test_table table. If it does not, the system reports an error and stops the upload.
```

- **download:** downloads data from a MaxCompute table. You can download data from only one table or partition to a file each time. For a partitioned table, you must specify the partition from which you want to download data. If the table has multiple levels of partitions, you must specify the last-level partition.

```
tunnel download test_project.test_table/p1="b1",p2="b2" test_table.txt;
-- Download data from the test_project.test_table table that has two levels of partitions to the test_table.txt file.
```

- **resume:** resumes the transfer of files or directories. The transfer is interrupted because your network is disconnected or Tunnel is faulty. You can use this command to resume only data uploads. One data download or upload is referred to as a session. You must specify the session ID in the RESUME command before you run this command.

```
tunnel resume;
```

- **show:** shows historical task information.

```
tunnel show history -n 5;
-- Show the commands used in the last five data uploads or downloads.
tunnel show log;
-- Show the logs of the last data upload or download.
```

- **purge:** clears a session directory. By default, sessions from the last three days are cleared.

```
tunnel purge 5;
--Clear logs from the last five days.
```

- **help:** obtains help information.

Upload

Description: This command is used to upload data to a MaxCompute table in append mode.

You can run the `tunnel help upload;` subcommand to query help information. The following result is returned:

```
usage: tunnel upload [options] <path> <[project.]table[/partition]>
      upload data from local file
-acp,-auto-create-partition <ARG> auto create target partition if not
      exists, default false
-bs,-block-size <ARG>          block size in MiB, default 100
-c,-charset <ARG>              specify file charset, default ignore.
      set ignore to download raw data
-cf,-csv-format <ARG>          use csv format (true|false), default
      false. When uploading in csv format,
      file splitting not supported.
-cp,-compress <ARG>            compress, default true
-dbr,-discard-bad-records <ARG> specify discard bad records
      action(true|false), default false
-dfp,-date-format-pattern <ARG> specify date format pattern, default
      yyyy-MM-dd HH:mm:ss
-fd,-field-delimiter <ARG>     specify field delimiter, support
      unicode, eg \u0001. default ", "
-h,-header <ARG>               if local file should have table
      header, default false
-mbr,-max-bad-records <ARG>    max bad records, default 1000
-ni,-null-indicator <ARG>     specify null indicator string,
      default ""(empty string)
-ow,-overwrite <true | false>  overwrite specified table or
      partition, default: false
-rd,-record-delimiter <ARG>    specify record delimiter, support
      unicode, eg \u0001. default "\r\n"
-s,-scan <ARG>                 specify scan file
      action(true|false|only), default true
-sd,-session-dir <ARG>        set session dir, default
      D:\software\odpscmd_public\plugins\dship
-ss,-strict-schema <ARG>      specify strict schema mode. If false,
      extra data will be abandoned and
      insufficient field will be filled
      with null. Default true
-t,-threads <ARG>              number of threads, default 1
-te,-tunnel_endpoint <ARG>     tunnel endpoint
-time,-time <ARG>              keep track of upload/download elapsed
      time or not. Default false
-tz,-time-zone <ARG>           time zone, default local timezone:
      Asia/Shanghai
```

Example:
tunnel upload log.txt test_project.test_table/p1="b1",p2="b2"

The following table describes the parameters.

Parameter	Description
path	The path and name of the file that you want to upload.
[project.]table[/partition]	The name of the table to which you want to upload data. You must specify the last-level partition for a partitioned table. If the table does not belong to the current project, you must specify the project where the table is located.
-acp	The partition to which you want to upload data. If the specified partition does not exist, a partition is automatically created. Default value: False.

Parameter	Description
-bs	The size of the data block uploaded by Tunnel each time. Default value: 100 MiB. 1 MiB = 1024 × 1024 bytes.
-c	The encoding format of the data file. The default value is UTF-8 without timing. The source data is downloaded by default.
-cf	Specifies whether the file is a CSV file. Default value: False.
-cp	Specifies whether to compress the file before you upload it to MaxCompute to reduce network traffic. Default value: True.
-dbr	Specifies whether to omit dirty data, such as additional columns, missing columns, or unmatched types of column data. The value True indicates that all data that does not match the schema of the table is omitted. The value False indicates that an error is returned when dirty data is detected. This ensures that raw data in the table to which you want to upload data is not contaminated. Default value: False.
-dfp	The format of DATETIME data. The default format is yyyy-MM-dd HH:mm:ss. If you want to specify the DATETIME data that is accurate to milliseconds, use <code>tunnel upload -dfp 'yyyy-MM-dd HH:mm:ss.SSS'</code> .
-fd	The column delimiter for the data file. The default value is a comma (,).
-h	Specifies whether the data file has table headers. If this parameter is set to True, the data from the second row in the file is uploaded. Default value: False.
-mbr	The maximum number of dirty data records. If the number of dirty data records exceeds the value of this parameter, the upload stops. Default value: 1000.
-ni	The NULL data identifier. The default value is an empty string.
-ow	Specifies whether the uploaded data overwrites the table or partition. Default value: False, which indicates that data is appended.
-rd	The row delimiter for the data file. The default value is \n in a Linux operating system or \r\n in a Windows operating system.
-s	Specifies whether to scan the data file. Default value: True. The value True indicates that the system scans the data and starts to import it only if the data is in the correct format. The value False indicates that the system imports data without scanning. The value Only indicates that the system only scans the data but does not import it.
-sd	The path of the session directory. Default value: <code>D:/console/plugins/dship/lib/...</code>
-ss	The strict schema mode. Default value: True. If the parameter is set to False, extra data is discarded and the fields that are not specified are filled with NULL.
-t	The number of threads. Default value: 1.
-te	The endpoint of Tunnel.

Parameter	Description
-time	Specifies whether the upload time is tracked. Default value: False.
-tz	The time zone. The local time zone is used by default. Example: Asia/Shanghai.

Examples:

1. Execute the following statement to create a partitioned table named sale_detail:

```
CREATE TABLE IF NOT EXISTS sale_detail(
  shop_name STRING,
  customer_id STRING,
  total_price DOUBLE)
PARTITIONED BY (sale_date STRING,region STRING);
```

2. Execute the following statement to add a partition to the sale_detail table:

```
alter table sale_detail add partition (sale_date='201705', region='hangzhou');
```

3. Prepare the data.txt file that contains the following data and save the file in the d:\data.txt directory:

```
shop9,97,100
shop10,10,200
shop11,11
```

 **Note** In this example, the third row of the data.txt file does not comply with the schema of the sale_detail table. The sale_detail table defines three columns, but the third row of this file contains only two columns.

4. Run the following command to upload the data.txt file to the sale_detail table.

```
tunnel u d:\data.txt sale_detail/sale_date=201705,region=hangzhou -s false;
```

The data upload fails because the data.txt file contains dirty data. The system returns the session ID and an error message.

```
Upload session: 201706101639224880870a002ec60c
Start upload:d:\data.txt
Total bytes:41 Split input to 1 blocks
2017-06-10 16:39:22 upload block: '1'
ERROR: column mismatch -,expected 3 columns, 2 columns found, please check data or delimiter
```

5. Execute the following statement to verify the data:

```
select * from sale_detail where sale_date='201312';
```

The data upload fails because dirty data is detected. As a result, the sale_detail table is empty.

```
ID = 20150610xxxxxxxxxxxxvc61z5
+-----+-----+-----+-----+-----+
| shop_name | customer_id | total_price | sale_date | region |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

6. Run the following command to query the ID of the failed session.

```
tunnel show history;
```

The following result is returned:

```
20150610xxxxxxxxxx70a002ec60c failed 'u --config-file /D:/console/conf/odps_config.ini --project odpstest_ay52c_ay52 --endpoint http://service.cn-shanghai.maxcompute.aliyun.com/api --id Uxxxxxxxxxxr11 --key 2m4r3WvTxxxxxxxx0InVke7UkvR d:\data.txt sale_detail/sale_date=201312,region=hangzhou -s false'
```

7. Modify the data.txt file to comply with the schema of the sale_detail table.

```
shopx,x_id,100  
shopy,y_id,200
```

8. Run the RESUME command to resume the data upload. In the command, 20150610xxxxxxxxxx70a002ec60c is the ID of the failed session.

```
tunnel resume 20150610xxxxxxxxxx70a002ec60c --force;
```

The following result is returned:

```
start resume  
20150610xxxxxxxxxx70a002ec60c  
Upload session: 20150610xxxxxxxxxx70a002ec60c  
Start upload:d:\data.txt  
Resume 1 blocks  
2015-06-10 16:46:42 upload block: '1'  
2015-06-10 16:46:42 upload block complete, blockid=1  
upload complete, average speed is 0 KB/s  
OK
```

9. Execute the following statement to verify the data:

```
select * from sale_detail where sale_date='201312';
```

If the following result is returned, the upload succeeded:

```
ID = 20150610xxxxxxxxxxa741z5  
+-----+-----+-----+-----+-----+  
| shop_name | customer_id | total_price | sale_date | region |  
+-----+-----+-----+-----+-----+  
| shopx | x_id | 100.0 | 201312 | hangzhou |  
| shopy | y_id | 200.0 | 201312 | hangzhou |  
+-----+-----+-----+-----+-----+
```

Download

Description: This command is used to download MaxCompute table data or the execution result of a specific instance to your computer.

You can run the `tunnel help download;` subcommand to query help information. The following result is returned:

```
Usage: tunnel download [options] <[project.]table[/partition]> <path>  
download data to local file  
-c,-charset <ARG> specify file charset, default ignore.  
-i,-ignore set ignore to download raw data  
-cf,-csv-format <ARG> use csv format (true|false), default  
false. When uploading in csv format,  
file splitting not supported.
```

```

-ci,-columns-index <ARG>    specify the columns index(starts from
                             0) to download, use comma to split each
                             index
-cn,-columns-name <ARG>     specify the columns name to download,
                             use comma to split each name
-cp,-compress <ARG>        compress, default true
-dfp,-date-format-pattern <ARG> specify date format pattern, default
                             yyyy-MM-dd HH:mm:ss
-e,-exponential <ARG>      When download double values, use
                             exponential express if necessary.
                             Otherwise at most 20 digits will be
                             reserved. Default false
-fd,-field-delimiter <ARG>  specify field delimiter, support
                             unicode, eg \u0001. default ", "
-h,-header <ARG>           if local file should have table header,
                             default false
-limit <ARG>               specify the number of records to
                             download
-ni,-null-indicator <ARG>  specify null indicator string, default
                             ""(empty string)
-rd,-record-delimiter <ARG> specify record delimiter, support
                             unicode, eg \u0001. default "\r\n"
-sd,-session-dir <ARG>     set session dir, default
                             D:\software\odpscmd_public\plugins\dship
-t,-threads <ARG>          number of threads, default 1
-te,-tunnel_endpoint <ARG> tunnel endpoint
-time,-time <ARG>          keep track of upload/download elapsed
                             time or not. Default false
-tz,-time-zone <ARG>       time zone, default local timezone:
                             Asia/Shanghai
usage: tunnel download [options] instance://<[project/]instance_id> <path>
       download instance result to local file
-c,-charset <ARG>          specify file charset, default ignore.
                             set ignore to download raw data
-cf,-csv-format <ARG>      use csv format (true|false), default
                             false. When uploading in csv format,
                             file splitting not supported.
-ci,-columns-index <ARG>    specify the columns index(starts from
                             0) to download, use comma to split each
                             index
-cn,-columns-name <ARG>     specify the columns name to download,
                             use comma to split each name
-cp,-compress <ARG>        compress, default true
-dfp,-date-format-pattern <ARG> specify date format pattern, default
                             yyyy-MM-dd HH:mm:ss
-e,-exponential <ARG>      When download double values, use
                             exponential express if necessary.
                             Otherwise at most 20 digits will be
                             reserved. Default false
-fd,-field-delimiter <ARG>  specify field delimiter, support
                             unicode, eg \u0001. default ", "
-h,-header <ARG>           if local file should have table header,
                             default false
-limit <ARG>               specify the number of records to
                             download
-ni,-null-indicator <ARG>  specify null indicator string, default
                             ""(empty string)
-rd,-record-delimiter <ARG> specify record delimiter, support

```

```

    unicode, eg \u0001. default "\r\n"
-sd,-session-dir <ARG>      set session dir, default
                             D:\software\odpscmd_public\plugins\dshi
-t,-threads <ARG>          number of threads, default 1
-te,-tunnel_endpoint <ARG>  tunnel endpoint
-time,-time <ARG>          keep track of upload/download elapsed
                             time or not. Default false
-tz,-time-zone <ARG>       time zone, default local timezone:
                             Asia/Shanghai
Example:
tunnel download test_project.test_table/p1="b1",p2="b2" log.txt // Download data from a specific table.
tunnel download instance://test_project/test_instance log.txt // Download the execution result of a specific instance.
  
```

The following table describes the parameters.

Parameter	Description
path	The path in which the downloaded data file is saved.
[project.]table[/partition]	The name of the table that you want to download. You must specify the last-level partition for a partitioned table. If the table does not belong to the current project, you must specify the project where the table is located.
[project/]instance_id	The ID of the instance. You must specify this parameter to download the execution result of a specific instance.
-fd	The column delimiter for the data file. The default value is a comma (,).
-rd	The row delimiter for the data file. Default value: \r\n.
-dfp	The format of DATETIME data. The default format is yyyy-MM-dd HH:mm:ss.
-ni	The NULL data identifier. The default value is an empty string.
-c	The encoding format of the data file. Default value: UTF-8.
-cf	Specifies whether the file is a CSV file. Default value: False.
-ci	Specifies that column indexes (starting from 0) are downloaded. Separate column indexes with commas (,).
-cp	The name of the column to download. Separate multiple column names with commas (,).
-e	Specifies whether the data of the DOUBLE type you want to download is represented by an exponential function if required. If the data is not represented by an exponential function, a maximum of 20 bits are retained. Default value: False.
-h	Specifies whether the data file has table headers. Default value: False. If this parameter is set to True, the data from the second row in the file is downloaded. <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note -h=true and threads>1 cannot be used together. threads>1 indicates multiple threads.</p> </div>
-limit	The number of files you want to download.

Parameter	Description
-tz	The time zone. The local time zone is used by default. Example: Asia/Shanghai.
-time	Specifies whether the upload time is tracked. Default value: False.
-te	The endpoint of Tunnel.
-t	The number of threads. Default value: 1.
-sd	The path of the session directory. Default value: <i>D:/console/plugins/dship/lib/...</i>

Examples:

1. Execute the following statement to query the sale_detail table:

```
select * from sale_detail;
```

The following result is returned:

```
ID = 20170724071705393ge3csfb8
```

```
... ..
```

2. Run the following command to download the execution results to your computer:

```
tunnel download instance://20170724071705393ge3csfb8 result;
```

If the following result is returned, the download succeeded:

```
2017-07-24 15:18:47 - new session: 2017072415184785b6516400090ca8 total lines: 8
2017-07-24 15:18:47 - file [0]: [0, 8), result
downloading 8 records into 1 file
2017-07-24 15:18:47 - file [0] start
2017-07-24 15:18:48 - file [0] OK. total: 44 bytes
download OK
```

After you enable use_instance_tunnel on the MaxCompute client, you can use InstanceTunnel to download the query results that are returned by SELECT statements. This helps you download query results of any size at any time. You can use one of the following methods to enable this feature:

- On the client, set use_instance_tunnel to true in the odps_config.ini file and make sure that instance_tunnel_max_record is automatically set to 10000.

```
# download sql results by instance tunnel
use_instance_tunnel=true
# the max records when download sql results by instance tunnel
instance_tunnel_max_record=10000
```

Note The instance_tunnel_max_record parameter specifies the maximum number of SQL result records that can be downloaded by using InstanceTunnel. If you do not specify this parameter, the number of result records that can be downloaded is not limited.

- Set console.sql.resultinstancetunnel to true.

```
set console.sql.resultinstancetunnel=true;
```

Resume

Description: This command is used to resume historical operations. Only data uploads can be resumed.

You can run the `tunnel help resume;` subcommand to query help information. The following result is returned:

```
Usage: tunnel resume [session_id] [--force]
       resume an upload session
-f,-force force resume
Example:
tunnel resume
```

Parameters:

- `session_id`: the session ID of the failed data upload. You must specify this parameter.
- `-f`: specifies whether to forcibly resume historical operations. By default, this parameter is not specified.

Examples:

Run the RESUME command to resume the session for which the upload failed. In this command, 20150610xxxxxxxxxx70a002ec60c is the ID of the session for which the upload failed.

```
tunnel resume 20150610xxxxxxxxxx70a002ec60c -force;
```

The following result is returned:

```
start resume
201706101639224880870a002ec60c
Upload session: 201706101639224880870a002ec60c
Start upload:d:\data.txt
Resume 1 blocks
2017-06-10 16:46:42 upload block: '1'
2017-06-10 16:46:42 upload block complete, blockid=1
upload complete, average > speed is 0 KB/s
OK
```

Show

Description: This command is used to show historical records.

You can run the `tunnel help show;` subcommand to query help information. The following result is returned:

```
Usage: tunnel show history [options]
       show session information
-n,-number <ARG> lines
Example:
tunnel show history -n 5
tunnel show log
```

Parameters: `-n` specifies the number of rows to be displayed.

Purge

Description: This command is used to clear a session directory.

You can run the `tunnel help purge;` subcommand to query help information. The following result is returned:

```
usage: tunnel purge [n]
       force session history to be purged.([n] days before, default
       3 days)
Example:
tunnel purge 5
```

Parameters: n specifies after how many days historical logs are cleared. Default value: 3.

1.5.2.9. Other operations

This topic describes common statements for other operations.

ALIAS

The ALIAS statement is used to create an alias for a resource. You can create the same alias for different resources to read these resources from the MaxCompute MapReduce or UDF code without code modification.

Syntax:

```
ALIAS <alias>=<real>;
```

Description: This statement is used to create an alias for a resource.

Parameters:

- alias: the alias of the resource.
- real: the original name of the resource.

Examples:

```
ADD TABLE src_part PARTITION (ds='20171208') AS res_20171208;
ADD TABLE src_part PARTITION (ds='20171209') AS res_20171209;
-- Add the res_20171208 and res_20171209 resources.
ALIAS resName=res_20171208;
jar -resources resName -libjars work.jar -classpath ./work.jar com.company.MainClass args ... ;
-- Set the alias of the res_20171208 resource to resName and call this resource.
ALIAS resName=res_20171209;
jar -resources resName -libjars work.jar -classpath ./work.jar com.company.MainClass args ... ;
-- Set the alias of the res_20171209 resource to resName and call this resource.
```

 **Note** In the preceding example, different resource tables are referenced by the same resource alias resName in two jobs. Different data is read with the same code.

Set

Syntax:

```
set <KEY>=<VALUE>
```

Description: This statement is used to configure built-in or user-defined system variables of MaxCompute to affect MaxCompute execution rules.

Parameters:

- KEY: the name of the attribute that you want to set.
- VALUE: the value of the attribute.

MaxCompute SQL and the latest version of MaxCompute MapReduce support the following SET statements:

```
set odps.stage.mapper.mem=  
-- Set the memory size of each mapper. Unit: MB. Default value: 1024.  
set odps.stage.reducer.mem=  
-- Set the memory size of each reducer. Unit: MB. Default value: 1024.  
set odps.stage.joiner.mem=  
-- Set the memory size of each joiner. Unit: MB. Default value: 1024.  
set odps.stage.mem=  
-- Set the total memory size of all workers in a specific MaxCompute job. This statement has a lower priority than the preceding three statements. The value is in MB. No default value is defined.  
set odps.stage.mapper.split.size=  
-- Set the input data volume of each mapper to indirectly control the number of workers in each map stage. The input data volume indicates the size of each slice in the input file. Unit: MB. Default value: 256.  
set odps.stage.reducer.num=  
-- Set the number of workers in each reduce stage. No default value is defined.  
set odps.stage.joiner.num=  
-- Set the number of workers in each join stage. No default value is defined.  
set odps.stage.num=  
-- Modifies the worker concurrency at all stages in a specific MaxCompute job. This statement has a lower priority than the preceding three statements. No default value is defined.
```

The early versions of MaxCompute MapReduce support the following SET statements:

```
set odps.mapred.map.memory=  
-- Set the memory size of each mapper. Unit: MB. Default value: 1024.  
set odps.mapred.reduce.memory=  
-- Set the memory size of each reducer. Unit: MB. Default value: 1024.  
set odps.mapred.map.split.size=  
-- Set the input data volume of each mapper to indirectly control the number of workers in each map stage. The input data volume indicates the size of each slice in the input file. Unit: MB. Default value: 256.  
set odps.mapred.reduce.tasks=  
-- Set the number of workers in each reduce stage. No default value is defined.
```

Examples:

- Adjust the cache size pre-defined for a complex column when data is written to MaxCompute tables to improve write performance.

```
set odps.sql.executionengine.coldata.deep.buffer.size.max=1048576;
```

- Set the size of data read by each mapper to 256 MB.

```
set odps.stage.mapper.split.size=256;
```

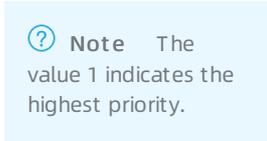
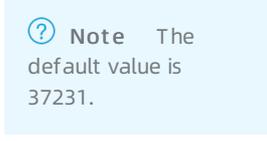
SetProject

Syntax:

```
setproject ["<KEY>=<VALUE>"];
```

Description: This statement is used to configure project attributes. If <KEY>=<VALUE> is not specified, the current attribute configurations of the project are displayed.

The following table describes project attributes.

Attribute	Permission owner	Description	Valid value
odps.table.drop.ignorenonexistent	All users	Indicates whether to report an error when you try to delete a table that does not exist. If the value is True, no error is reported.	True and False
odps.instance.priority.autoadjust	Project owner	Indicates whether to automatically prioritize small tasks.	True and False
odps.instance.priority.level	Project owner	Indicates the priority of a task in a project.	1 to 3 
odps.security.ip.whitelist	Project owner	Indicates an IP address whitelist for the project.	List of IP addresses separated by commas (,)
odps.table.lifecycle	Project owner	optional: The LIFECYCLE clause is optional in the statement that is used to create a table. If no lifecycle is set for a table, the table does not expire. mandatory: The LIFECYCLE clause is required. inherit: If no lifecycle is set for a table, the value of odps.table.lifecycle.value is the lifecycle of this table.	optional, mandatory, and inherit
odps.table.lifecycle.value	Project owner	Indicates the default lifecycle.	1 to 37231 
odps.instance.retain.days	Project owner	Indicates the retention period for instance information, in days.	3 to 30
odps.task.sql.outerjoin.pushdown	Project owner	Indicates whether the filter conditions in FULL OUTER JOIN are pushed down.	True and False

Attribute	Permission owner	Description	Valid value
odps.function.strictmode	Project owner	Indicates whether to return NULL or report an error when dirty data is involved in the process of using built-in functions. The value False indicates that NULL is returned. The value True indicates that an error is returned.	True and False
odps.task.sql.write.strnull	Project owner	Indicates whether to consider empty strings as NULL. If the value is True, empty strings are considered NULL.	True and False

EXPORT

Syntax:

```
export <projectname> <local_path>;
```

Description: This statement is used to export the metadata of a project to your computer. Metadata is represented by statements acceptable by the MaxCompute client (odpscmd). The exported metadata file can be used to recreate a project.

Show Flags

Syntax:

```
show flags;
```

Description: This statement is used to show the parameters configured by using SET statements.

 **Note** This statement takes effect only at the project level.

COST SQL

Syntax:

```
cost sql <SQL Sentence>;
```

Description: This statement is used to estimate the costs of an SQL statement based on the size of the input data, the number of UDFs, and the SQL complexity.

Examples:

Execute the following statement to estimate the costs of an SQL statement :

```
cost sql select distinct project_name, user_name from meta.m_security_users distribute by project_name sort by project_name;
```

The following result is returned:

```
ID = 20190715113033121xxxxxxx
Input:65727592 Bytes
UDF:0
Complexity:1.0
```

1.6. MaxCompute SQL

1.6.1. Overview

1.6.1.1. Scenarios

This topic describes the scenarios of MaxCompute SQL.

MaxCompute SQL offline computing is applicable to scenarios where you need to process terabytes of data but do not require real-time processing. It requires a relatively long time to prepare and submit jobs for MaxCompute SQL offline computing. Therefore, MaxCompute SQL is not well-suited for scenarios where thousands or even tens of thousands of transactions need to be processed per second. MaxCompute SQL online computing provides near real-time (NRT) processing capabilities.

The MaxCompute SQL syntax is similar to the SQL syntax. The MaxCompute SQL syntax can be considered a subset of standard SQL. MaxCompute SQL lacks many database features, such as transactions, primary key constraints, and indexes. Therefore, MaxCompute SQL cannot be regarded as a database. The maximum size of each SQL statement supported by MaxCompute is 2 MB.

1.6.1.2. Reserved words

The keywords of SQL statements are reserved words in MaxCompute. Do not use the reserved words when you name tables, columns, or partitions. If you use the reserved words, an error is reported. Reserved words are not case-sensitive.

This section lists the common reserved words. For a complete list of reserved words, see [Reserved words](#).

```
% && ( ) * + - . / ; < <= <>
=> >= ? ADD ALL ALTER
AND AS ASC BETWEEN BIGINT BOOLEAN BY
CASE CAST COLUMN COMMENT CREATE DESC DISTINCT
DISTRIBUTE DOUBLE DROP ELSE FALSE FROM FULL
GROUP IF IN INSERT INTO IS JOIN
LEFT LIFECYCLE LIKE LIMIT MAPJOIN NOT NULL
ON OR ORDER OUTER OVERWRITE PARTITION RENAME
REPLACE RIGHT RLIKE SELECT SORT STRING TABLE
THEN TOUCH TRUE UNION VIEW WHEN WHERE
```

1.6.1.3. Partitioned tables

This topic describes the advantages and disadvantages of MaxCompute partitioned tables.

You can specify partition key columns of a partitioned table as filter conditions in WHERE clauses of SELECT statements. This improves SQL query performance and reduces your costs. However, some SQL statements for partition operations are less efficient. For example, if more than 10,000 partitions exist in an output table for a job, the SQL statements may fail.

If you use a partitioned table, you must properly evaluate the number and levels of partitions based on the data source to ensure query performance and stability. MaxCompute allows a maximum of six levels of partitions in a partitioned table.

For some statements, the syntax differs between partitioned and non-partitioned tables. For more information, see [DDL statements](#) and [DML statements](#).

For more information about the statements that are used to create tables, see [Create a table](#).

1.6.1.4. Type conversion

1.6.1.4.1. Explicit conversions of data types

An explicit conversion uses `CAST` to convert a value type into another one. This topic describes explicit conversions.

For more information about `CAST`, see [CAST](#).

The following table lists the explicit conversions supported by MaxCompute SQL.

Explicit conversions

From/To	BIGINT	DOUBLE	STRING	DATETIME	BOOLEAN	DECIMAL
BIGINT	N/A	Y	Y	N	N	Y
DOUBLE	Y	N/A	Y	N	N	Y
STRING	Y	Y	N/A	Y	N	Y
DATETIME	N	N	Y	N/A	N	N
BOOLEAN	N	N	N	N	N/A	N
DECIMAL	Y	Y	Y	N	N	N/A

Y indicates that the conversion is supported. N indicates that the conversion is not supported. N/A indicates that the conversion is not required.

 Note

- If the DOUBLE type is converted into the BIGINT type, the fractional part is truncated. Example: `cast(1.6 as bigint) = 1`.
- If the STRING type that meets the format of the DOUBLE type is converted into the BIGINT type, the STRING type is first converted into the DOUBLE type and then to the BIGINT type. Therefore, the fractional part is truncated. Example: `cast("1.6" as bigint) = 1`.
- If the STRING type that meets the format of the BIGINT type is converted into the DOUBLE type, one digit is retained after the decimal point. Example: `cast("1" as double) = 1.0`.
- To convert a constant string into the DECIMAL type, enclose the constant string within a pair of double quotation marks ("). If the value is not enclosed in quotation marks, it is considered a DOUBLE-type value. Example: `cast("1.234567890123456789" as decimal)`.
- An unsupported explicit conversion causes an exception.
- If a conversion fails, the system returns an error and exits.
- The conversion of the DATETIME type uses the default format `yyyy-mm-dd hh:mi:ss:ff3`. For more information, see [Conversion between string and datetime types](#).
- Some data types cannot be explicitly converted, but can be converted by using SQL built-in functions. For example, you can use the `TO_CHAR` function to convert the BOOLEAN type into the STRING type. For more information, see [TO_CHAR](#). You can also use the `TO_DATE` function to convert the STRING type into the DATETIME type.
- If the values of the DECIMAL type are not within the value range, the cast string to decimal operation may return an error, such as most significant bit overflow or least significant bit overflow truncation.

1.6.1.4.2. Implicit type conversion and its scope

An implicit type conversion allows MaxCompute to automatically convert data types based on the context and predefined rules. This topic describes the rules of implicit conversions.

The following table lists the rules of implicit conversions supported by MaxCompute.

Implicit conversion 1

From/To	BOOLEAN	TINYINT	SMALLINT	INT	BIGINT	FLOAT
BOOLEAN	T	F	F	F	F	F
TINYINT	F	T	T	T	T	T
SMALLINT	F	F	T	T	T	T
INT	F	F	F	T	T	T
BIGINT	F	F	F	F	T	T
FLOAT	F	F	F	F	F	T
DOUBLE	F	F	F	F	F	F
DECIMAL	F	F	F	F	F	F
STRING	F	F	F	F	F	F

From/To	BOOLEAN	TINYINT	SMALLINT	INT	BIGINT	FLOAT
VARCHAR	F	F	F	F	F	F
TIMESTAMP	F	F	F	F	F	F
BINARY	F	F	F	F	F	F

Implicit conversion 2

From/To	DOUBLE	DECIMAL	STRING	VARCHAR	TIMESTAMP	BINARY
BOOLEAN	F	F	F	F	F	F
TINYINT	T	T	T	T	F	F
SMALLINT	T	T	T	T	F	F
INT	T	T	T	T	F	F
BIGINT	T	T	T	T	F	F
FLOAT	T	T	T	T	F	F
DOUBLE	T	T	T	T	F	F
DECIMAL	F	T	T	T	F	F
STRING	T	T	T	T	F	F
VARCHAR	T	T	T	T	F	F
TIMESTAMP	F	F	T	T	T	F
BINARY	F	F	F	F	F	T

T indicates that the conversion is supported. F indicates that the conversion is not supported.

Note

- If an unsupported implicit conversion is performed, an error is returned.
- If a conversion fails, the system returns an error and exits.
- MaxCompute automatically performs implicit conversions on data types based on the context. If the types do not match, you can use the CAST function to explicitly convert data types.
- The rules of implicit conversions apply to specific scopes. In specific scenarios, only some rules take effect.

Implicit conversions with relational operators

Relational operators include =, <>, <, ≤, >, ≥, IS NULL, IS NOT NULL, LIKE, RLIKE, and IN. The rules for implicit conversions that use LIKE, RLIKE, and IN are different from those that use other relational operators. The rules described in this section do not apply to the three operators. The following table lists the rules of implicit conversions when relational operations are used for data of different types.

Implicit conversions with relational operators

From/To	BIGINT	DOUBLE	STRING	DATETIME	BOOLEAN	DECIMAL
BIGINT	N/A	DOUBLE	DOUBLE	N	N	DECIMAL
DOUBLE	DOUBLE	N/A	DOUBLE	N	N	DECIMAL
STRING	DOUBLE	DOUBLE	N/A	DATETIME	N	DECIMAL
DATETIME	N	N	DATETIME	N/A	N	N
BOOLEAN	N	N	N	N	N/A	N
DECIMAL	DECIMAL	DECIMAL	DECIMAL	N	N	N/A

 **Note**

- If an implicit conversion is not supported between two values that you want to compare, the relational operation cannot be performed and an error is returned.
- For more information about relational operators, see [Relational operators](#).

Implicit conversion with special relational operators

Special relational operators include LIKE, RLIKE, and IN.

Syntax of LIKE and RLIKE:

```
source like pattern;
source rlike pattern;
```

Take note of the following points for the two relational operators in implicit conversions:

- The source and pattern parameters of LIKE and RLIKE must be of the STRING type.
- Other types can neither be involved in the operation nor be implicitly converted into the STRING type.
- If the value of source or pattern is NULL, NULL is returned.

Syntax of IN:

```
key in (value1, value2,...);
```

Rules of implicit conversions for IN:

- The data types of the values on the right of the IN keyword must be consistent.
- MaxCompute automatically compares the values in the column specified by key and values on the right of the IN keyword. If these values are of the BIGINT, DOUBLE, and STRING types, the values are converted into the DOUBLE type for comparison. If these values are of the DATETIME and STRING types, the values are converted into the DATETIME type for comparison. Conversions between other types are not allowed.

 **Note** The memory used by the compiler increases with the number of parameters used in the IN operation. If 5,000 parameters are used in an IN operation, the GCC compiler consumes 17 GB of memory for compilation. We recommend that you limit the number of parameters to 1,024. In this case, the compiler consumes a maximum of 1 GB of memory and requires 39 seconds for compilation.

Implicit conversions with arithmetic operators

Arithmetic operators include addition (+), subtraction (-), multiplication (×), division (/), and percent (%). This section describes the rules for implicit conversions that use these operators.

- Only the STRING, BIGINT, DECIMAL, and DOUBLE types can be used in arithmetic operations.
- Before an arithmetic operation, STRING data is implicitly converted into DOUBLE data.
- If an arithmetic operation involves data of the BIGINT and DOUBLE types, BIGINT data is implicitly converted into DOUBLE data.
- The DATETIME and BOOLEAN types cannot be used in arithmetic operations.

 **Note** For more information about arithmetic operators, see [Arithmetic operators](#).

Implicit conversions with logical operators

Logical operators include AND, OR, and NOT. This section describes the rules for implicit conversions that use these operators.

- Only the BOOLEAN type can be used in logical operations.
- The other types are not supported by logical operations or implicit conversions.

 **Note** For more information about logical operators, see [Logical operators](#).

1.6.1.4.3. SQL built-in functions

MaxCompute SQL provides a variety of built-in functions. These functions can be used to calculate one or more columns of a row and provide data of any type.

The following rules apply to implicit conversions:

- If you call a function and the data type of an input parameter is different from that defined in the function, the data type of the input parameter is converted into the function-defined data type.
- The parameters of each built-in function in MaxCompute SQL have different requirements for implicit conversions. For more information, see [Built-in functions](#).

1.6.1.4.4. CASE WHEN

This topic describes the implicit conversion rules for CASE WHEN.

Implicit conversion rules for CASE WHEN:

- If return values are of the BIGINT and DOUBLE types, all the values are converted into the DOUBLE type.
- If return values include those of the STRING type, all the values are converted into the STRING type. If a conversion, such as conversion from BOOLEAN into STRING, fails, an error is returned.
- Conversions between other types are not allowed.

1.6.1.4.5. Conversions between the STRING and DATETIME types

MaxCompute supports conversions between the STRING and DATETIME types.

The format used in conversions is yyyy-mm-dd hh:mi:ss.ff3.

Valid values of time units

Time unit	String (not case-sensitive)	Valid value
Year	yyyy	0001 to 9999
Month	mm	01 to 12
Day	dd	01 to 31
Hour	hh	00 to 23
Minute	mi	00 to 59
Second	ss	00 to 59
Millisecond	ff3	00 to 999

 **Note**

- If the first digit of a valid value for each time unit is 0, 0 cannot be omitted. For example, 2017-1-9 12:12:12 is an invalid DATETIME format and it cannot be converted from the STRING type into the DATETIME type. It must be written as 2017-01-09 12:12:12.
- Only the STRING type that meets the preceding format requirements can be converted into the DATETIME type. For example, `cast("2017-12-31 02:34:34" as datetime)` converts 2017-12-31 02:34:34 of the STRING type into the DATETIME type. Similarly, if the DATETIME type is converted into the STRING type, the default conversion format is yyyy-mm-dd hh:mi:ss. Conversions that use the following or similar statements may fail, and errors are returned.

```
cast("2017/12/31 02/34/34" as datetime)
cast("20171231023434" as datetime)
cast("2017-12-31 2:34:34" as datetime)
```

MaxCompute provides the `TO_DATE` function to convert the STRING type that does not meet the DATETIME format requirements into the DATETIME type. For more information, see [TO_DATE](#).

1.6.2. Operators

1.6.2.1. Relational operators

This topic describes relational operators in MaxCompute SQL operators.

Relational operators

Operator	Description
A=B	If A or B is NULL, NULL is returned. If A is equal to B, TRUE is returned. Otherwise, FALSE is returned.
A<>B	If A or B is NULL, NULL is returned. If A is unequal to B, TRUE is returned. Otherwise, FALSE is returned.
A<B	If A or B is NULL, NULL is returned. If A is less than B, TRUE is returned. Otherwise, FALSE is returned.
A<=B	If A or B is NULL, NULL is returned. If A is less than or equal to B, TRUE is returned. Otherwise, FALSE is returned.

Operator	Description
A > B	If A or B is NULL, NULL is returned. If A is greater than B, TRUE is returned. Otherwise, FALSE is returned.
A >= B	If A or B is NULL, NULL is returned. If A is greater than or equal to B, TRUE is returned. Otherwise, FALSE is returned.
A IS NULL	If A is NULL, TRUE is returned. Otherwise, FALSE is returned.
A IS NOT NULL	If A is not NULL, TRUE is returned. Otherwise, FALSE is returned.
A LIKE B	<p>If A or B is NULL, NULL is returned. A is a string and B is the pattern that you want to match. If A matches B, TRUE is returned. Otherwise, FALSE is returned. The percent sign (%) is a wildcard character that matches an arbitrary number of characters. The underscore (_) is a wildcard character that matches a single character. To use these two characters as regular characters, use backslashes (\) to escape them: \% and _.</p> <p>'aaa' like 'a' = TRUE 'aaa' like 'a%' = TRUE 'aaa' like 'aab' = FALSE 'a%b' like 'a\b' = TRUE 'axb' like 'a\b' = FALSE</p>
A RLIKE B	If A or B is NULL, NULL is returned. A is a string and B is a regular expression consisting of string constants. If A matches B, TRUE is returned. Otherwise, FALSE is returned. If B is an empty string, the system returns an error and exits.
A IN B	B is a set. If A is NULL, NULL is returned. If A is in B, TRUE is returned. Otherwise, FALSE is returned. If B contains only one element NULL, namely, A IN (NULL), NULL is returned. If B contains NULL, the type of NULL is considered the same as the other elements in B. B must be a constant and have at least one element. All elements must be of the same type.

The values of the DOUBLE type in MaxCompute are different in precision. We recommend that you do not use the equal sign (=) for comparison between two values of the DOUBLE type. You can deduct a value of the DOUBLE type from the other value of the DOUBLE type and use the absolute value of the difference to determine whether the two values are equal. If the absolute value is negligible, the two values of the DOUBLE type are considered equal. Example:

```
abs(0.9999999999 - 1.0000000000) < 0.0000000001
-- 0.9999999999 and 1.0000000000 have 10 decimal digits, whereas 0.0000000001 has 9 decimal digits.
-- 0.9999999999 is considered equal to 1.0000000000.
```

 **Note**

- ABS is a built-in function provided by MaxCompute to take the absolute value of its input. For more information, see [ABS](#).
- In most cases, a value of the DOUBLE type in MaxCompute can retain 16 valid digits.

1.6.2.2. Arithmetic operators

This topic describes arithmetic operators in MaxCompute SQL operators.

Arithmetic operators

Operator	Description
A + B	If A or B is NULL, NULL is returned. Otherwise, the result of A + B is returned.
A - B	If A or B is NULL, NULL is returned. Otherwise, the result of A - B is returned.
A * B	If A or B is NULL, NULL is returned. Otherwise, the result of A × B is returned.
A / B	If A or B is NULL, NULL is returned. Otherwise, the result of A/B is returned. If both A and B are of the BIGINT type, a value of the DOUBLE type is returned.
A % B	If A or B is NULL, NULL is returned. Otherwise, the result of A % B is returned.
+A	A is returned.
-A	If A is NULL, NULL is returned. Otherwise, -A is returned.

Note

- You can use only the values of the STRING, BIGINT, DOUBLE, or DECIMAL type to perform arithmetic operations. You cannot use the values of the DATETIME or BOOLEAN type to perform arithmetic operations.
- Values of the STRING type are implicitly converted into those of the DOUBLE type before arithmetic operations.
- If you use the values of the BIGINT and DOUBLE types to perform arithmetic operations, the value of the BIGINT type is implicitly converted into that of the DOUBLE type before arithmetic operations. A value of the DOUBLE type is returned.
- If both A and B are of the BIGINT type, a value of the DOUBLE type is returned after you perform the A/B operation. For other arithmetic operations, a value of the BIGINT type is returned.

1.6.2.3. Bitwise operators

This topic describes bitwise operators in MaxCompute SQL operators.

Bitwise operators

Operator	Description
A & B	The bitwise AND result of A and B is returned. For example, the result of 1&2 is 0, and the result of 1&3 is 1. The bitwise AND result of NULL and a value is NULL. Both A and B must be of the BIGINT type.
A B	The bitwise OR result of A and B is returned. For example, the result of 1 2 is 3, and the result of 1 3 is 3. The bitwise OR result of NULL and a value is NULL. Both A and B must be of the BIGINT type.

 **Notice** Bitwise operators do not support implicit type conversions. You can use the values only of the BIGINT type in bitwise operations.

1.6.2.4. Logical operators

This topic describes logical operators in MaxCompute SQL.

Logical operators

Operator	Description
A and B	TRUE and TRUE = TRUE
	TRUE and FALSE = FALSE
	FALSE and TRUE = FALSE
	FALSE and NULL = FALSE
	FALSE and FALSE = FALSE
	NULL and FALSE = FALSE
	TRUE and NULL = NULL
	NULL and TRUE = NULL
	NULL and NULL = NULL
A or B	TRUE or TRUE = TRUE
	TRUE or FALSE = TRUE
	FALSE or TRUE = TRUE
	FALSE or NULL = NULL
	NULL or FALSE = NULL
	TRUE or NULL = TRUE
	NULL or TRUE = TRUE
	NULL or NULL = NULL
NOT A	If A is NULL, NULL is returned.
	If A is TRUE, FALSE is returned.
	If A is FALSE, TRUE is returned.

 **Note** Logical operators do not support implicit type conversions. You can use only the values of the BOOLEAN type in logical operations.

1.6.3. LOAD

This topic describes how to use LOAD statements to import data from external storage, such as Object Storage Service (OSS), to a table or a table partition in MaxCompute.

Usage notes

MaxCompute must be authorized to access data in external storage. The methods used to authorize MaxCompute to access external data by using LOAD statements are the same as those used to authorize MaxCompute external tables. You can use one of the following methods for authorization:

- Specify the AccessKey ID and AccessKey secret in the directory in which data is stored. In this example, a directory on OSS is used.

```
'oss://<yourAccessKeyId>:<yourAccessKeySecret>@oss-cn-hangzhou-zmf.aliyuncs.com/my_bucket_id/my_location/'
```

 **Notice** If you use this method, the AccessKey ID and AccessKey secret are displayed in plaintext. This may pose security risks. We recommend that you do not use this method.

- Use Security Token Service (STS).

Use an extractor or a storage handler to import data

Syntax:

```
LOAD OVERWRITE|INTO TABLE table_name [PARTITION(partcol1=val1, partcol2=val2 ...)]
FROM LOCATION external_location
[STORED BY StorageHandler]
[WITH SERDEPROPERTIES (Options)];
```

Parameters:

- **table_name**: the name of the destination table into which you want to insert data. You must create a destination table before you insert data into it. The schema of the destination table must be the same as the format of the external data.
- **LOAD INTO**: inserts data into a table or partition.
- **LOAD OVERWRITE**: clears the existing data from a table or partition and inserts data into the table or partition.
- **SORTED BY**: specifies the name of the storage handler. You can use this clause in the same way as you use it for MaxCompute external tables.
- **LOCATION**: specifies the OSS directory in which the data files you want to read are saved. The system reads all files in the directory by default.
- **WITH SERDEPROPERTIES**: specifies the properties of the MaxCompute external table. The properties supported by WITH SERDEPROPERTIES are the same as those supported by the MaxCompute external table.

Assume that the vehicle.csv file contains the following data:

```
1,1,51,1,46.81006,-92.08174,9/14/2014 0:00,S
1,2,13,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,3,48,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,4,30,1,46.81006,-92.08174,9/14/2014 0:00,W
1,5,47,1,46.81006,-92.08174,9/14/2014 0:00,S
1,6,9,1,46.81006,-92.08174,9/14/2014 0:00,S
1,7,53,1,46.81006,-92.08174,9/14/2014 0:00,N
1,8,63,1,46.81006,-92.08174,9/14/2014 0:00,SW
1,9,4,1,46.81006,-92.08174,9/14/2014 0:00,NE
1,10,31,1,46.81006,-92.08174,9/14/2014 0:00,N
```

Examples:

```
oss://${accessKeyId}:${accessKeySecret}@oss-cn-hangzhou-zmf.aliyuncs.com/bucket/data_location/
-- Save the vehicle.csv file to a folder in an OSS bucket. Obtain the structure of the directory where the MaxCompute external table is saved.
CREATE TABLE ambulance_data_csv_load (
  vehicleId INT,
  recordId INT,
  patientId INT,
  calls INT,
  locationLatitude DOUBLE,
  locationLongitude DOUBLE,
  recordTime STRING,
  direction STRING );
-- Create a destination table named ambulance_data_csv_load.
LOAD OVERWRITE TABLE ambulance_data_csv_load
FROM
LOCATION 'oss://oss-cn-hangzhou-zmf.aliyuncs.com/bucket/data_location/'
STORED BY 'com.aliyun.odps.CsvStorageHandler'
WITH SERDEPROPERTIES (
  'odps.properties.rolearn'='acs:ram::xxxxx:role/aliyunodpsdefaultrole', -- The Alibaba Cloud Resource Name (ARN)
of the AliyunODPSDefaultRole role.
  'odps.text.option.delimiter'=',',
);
-- Import the CSV file from OSS to the destination table.
```

Import data stored in an open source format

Syntax:

```
LOAD OVERWRITE|INTO TABLE table_name [PARTITION(partcol1=val1, partcol2=val2 ...)]
FROM LOCATION external_location
[ROW FORMAT SERDE '<serde class>'
 [WITH SERDEPROPERTIES ('odps.properties.rolearn'='${roleran}' [, 'name2'='value2',...])]
]
STORED AS <file format>;
```

Parameters:

- **table_name**: the name of the destination table into which you want to insert data. You must create a destination table before you insert data into it. The schema of the destination table must be the same as the format of the external data.
- **LOAD INTO**: inserts data into a table or partition.
- **LOAD OVERWRITE**: clears the existing data from a table or partition and inserts data into the table or partition.
- **STORED AS**: specifies the format of the imported data file. Valid values include ORC, PARQUET, RCFILE, SEQUENCEFILE, and TEXT FILE. You can use this clause in the same way as you use it for MaxCompute external tables.
- **ROW FORMAT SERDE**: This clause is optional. You can use this clause in the same way as you use it for MaxCompute external tables. Mappings between file formats and SerDe classes:
 - SEQUENCEFILE: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
 - TEXT FILE: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
 - RCFILE: org.apache.hadoop.hive.serde2.columnar.LazyBinaryColumnarSerDe
 - ORC: org.apache.hadoop.hive.qi.io.orc.OrcSerde
 - ORCFILE: org.apache.hadoop.hive.qi.io.orc.OrcSerde

- PARQUET: org.apache.hadoop.hive ql.io.parquet.serde.ParquetHiveSerDe
- AVRO: org.apache.hadoop.hive.serde2.avro.AvroSerDe
- SERDEPROPERTIES: If you use STS to grant MaxCompute the required permissions, you must use this parameter to specify the value of the odps.properties.rolearn property. This property indicates the ARN of the AliyunODPSDefaultRole role. If the owner of MaxCompute and that of OSS use the same account, one-click authorization is allowed.

Examples:

```
oss://${accessKeyId}:${accessKeySecret}@oss-cn-hangzhou-zmf.aliyuncs.com/bucket/data_location/ds=20200910/'
-- Assume that the OSS directory where the MaxCompute external table is saved uses the preceding structure. The
MaxCompute external table in the directory does not include partition key columns. The information of partition ke
y columns is included in the directory information.
CREATE TABLE ambulance_data_csv_load_pt (
vehicleId STRING,
recordId STRING,
patientId STRING,
calls STRING,
locationLatitude STRING,
locationLongitude STRING,
recordTime STRING,
direction STRING)
PARTITIONED BY (ds STRING);
-- Create a destination table named ambulance_data_csv_load_pt.
LOAD OVERWRITE TABLE ambulance_data_csv_load_pt PARTITION(ds='20200910')
FROM
LOCATION 'oss://<yourAccessKeyId>:<yourAccessKeySecret>@oss-cn-hangzhou-zmf.aliyuncs.com/bucket/data_lo
cation/'
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
STORED AS TEXTFILE;
-- Import data stored in open source formats from OSS to the destination table.
```

 **Note** If you download data to a partitioned table, the schema of the external table in the OSS directory does not include partition key columns. In the preceding example, the external table in the directory contains only the business fields. The table does not include partition key columns.

1.6.4. DDL statements

1.6.4.1. Table operations

1.6.4.1.1. Create a table

This topic describes how to use a DDL statement to create a table.

Syntax:

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table_name
[(col_name data_type [DEFAULT value] [COMMENT col_comment], ...)]
[COMMENT table_comment]
[PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]
[CLUSTERED BY | RANGE CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...))] INTO number_of_buckets BUCKETS] -- Specify the shuffle and sort attributes when you create a clustered table.
[STORED BY StorageHandler] -- Used only for external tables.
[WITH SERDEPROPERTIES (Options)] -- Used only for external tables.
[LOCATION OSSLocation] -- Used only for external tables.
[STORED AS AliOrc]-- Specify the storage format of the table. You can specify AliORC only for newly created internal tables.
[TBLPROPERTIES("compressionstrategy"="normal/high/extreme",-- Specify the compression policy for data storage.
    "transactional"="true")] -- Specify a transactional table. You can then execute UPDATE or DELETE statements on the table.
[LIFECYCLE days];
CREATE TABLE [IF NOT EXISTS] table_name
[AS select_statement | LIKE existing_table_name];
```

Parameters:

- **table_name** and **col_name**: the table name and column name. The names are not case-sensitive. The names cannot contain special characters. A table or column name can contain only letters, digits, and underscores (_) and can be up to 128 bytes in length. We recommend that you start the table or column name with a letter. A table can contain a maximum of 1,200 columns.
- **data_type**: the type of data. Valid values: BIGINT, DOUBLE, BOOLEAN, DATETIME, DECIMAL, STRING, ARRAY<T>, and MAP<T1,T2>.

 **Note** If you want to use new data types, such as TINYINT, SMALLINT, INT, FLOAT, VARCHAR, TIMESTAMP, and BINARY, you must enable these data types.

- To enable new data types at the session level, add `set odps.sql.type.system.odps2=true;` before the SQL statement. Then, commit them for execution.
- To enable new data types at the project level, run the following command as the project owner:

```
setproject odps.sql.type.system.odps2=true;
```

- **DEFAULT value**: the default value of the column. If you do not specify the value of the column in an INSERT operation, the default value is used for the column.
- **COMMENT table_comment**: the comment of the table. The comment must be a valid string that can be up to 1,024 bytes in length.
- **PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)**: the partition field of the table. The following data types are supported: TINYINT, SMALLINT, INT, BIGINT, VARCHAR, and STRING.

A partition value cannot contain double-byte characters. It must start with a letter, followed by letters, digits, or special characters. The value can be up to 128 bytes in length. The value can contain the following special characters: spaces, colons (:), underscores (_), dollar signs (\$), number signs (#), periods (.), exclamation points (!), and at signs (@). Other characters, such as \t, \n, and /, are considered undefined characters. After you use partition fields to partition a table, a full table scan is not triggered when you add partitions, update partition data, or read partition data. This makes data processing more efficient.

 **Note** A table can contain a maximum of 60,000 partitions at six or lower levels.

- **CLUSTERED BY | RANGE CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...)] INTO number_of_buckets BUCKETS:** specifies the shuffle and sort attributes when you create a clustered table. Clustered tables are classified into hash-clustered tables and range-clustered tables.
 - Hash-clustered tables
 - **CLUSTERED BY:** the hash key. MaxCompute performs hash operations on the specified columns and distributes data to each bucket based on the hash values. We recommend that you specify a column that has large value ranges and a small number of duplicate key-value pairs in CLUSTERED BY. This prevents data skew and hot spots and makes concurrent execution more efficient. To optimize JOIN operations, we recommend that you select commonly used join or aggregation keys. The join and aggregation keys are similar to the primary keys in conventional databases.
 - **SORTED BY:** specifies how the data in a bucket is sorted. To achieve better performance, we recommend that you set SORTED BY and CLUSTERED BY to the same value. If you specify the SORTED BY clause, MaxCompute automatically generates indexes and uses these indexes to speed up your queries.
 - **INTO number_of_buckets BUCKETS:** the number of hash buckets. This parameter is required and varies based on your data volume. By default, MaxCompute supports a maximum of 1,111 reducers. Therefore, MaxCompute supports a maximum of 1,111 hash buckets. You can run the `set odps.sql.reducer.instances=xxx;` command to increase the number of hash buckets. However, the maximum number of hash buckets is 4,000. If the value exceeds 4000, computing performance deteriorates.

Note We recommend that you comply with the following rules when you specify the number of hash buckets:

- Keep the size of each bucket around 500 MB. For example, if you want to add 1,000 buckets to a partition whose size is 500 GB, the size of each bucket is 500 MB on average. If a table contains large amounts of data, you can increase the size of each bucket from 500 MB to a size in the range of 2 GB to 3 GB. You can also run the `set odps.sql.reducer.instances=xxx;` command to set the number of hash buckets to a value greater than 1111.
- To optimize JOIN operations, we recommend that you do not specify the shuffle and sort attributes. The number of hash buckets in one table must be the multiple of that in the other table. For example, one table has 256 hash buckets and the other table has 512 hash buckets. We recommend that you set the number of hash buckets to 2^n , such as 512, 1,024, 2,048, or 4,096. This way, MaxCompute can automatically split and merge hash buckets. To make the execution more efficient, we recommend that you do not specify the shuffle and sort attributes.

- Range-clustered tables
 - **RANGE CLUSTERED BY:** the column for range clustering. MaxCompute performs bucket operations on the specified columns and distributes data to each bucket based on the bucket numbers.
 - **SORTED BY:** specifies how the data in a bucket is sorted. You can use this clause in the same way you use it for a hash-clustered table.
 - **INTO number_of_buckets BUCKETS:** the number of hash buckets. Compared with hash-clustered tables, range-clustered tables have no limits on the number of buckets if the data is reasonably distributed. If you do not specify the number of buckets for a range-clustered table, MaxCompute automatically determines the optimal number of buckets based on your data volume.

Note If join and aggregate operations are performed on range-clustered tables and the join or group key is the range clustering key or the prefix of the range clustering key, you can control flags to remove data shuffling. This makes the execution more efficient. You can use the `set odps.optimizer.enable.range.partial.repartitioning` flag to control data shuffling. Valid values are true and false. The default value is false, which indicates that data shuffling is disabled.

- **STORED AS:** the storage format of the table. The default value is CFile2. AliORC in C++ is now available. It is developed by the MaxCompute storage team. AliORC is fully compatible with the open source Optimized Row Columnar (ORC). Compared with CFile2, AliORC frees up more than 10% of storage and improves read performance by more than 20%.
- **LIFECYCLE:** the lifecycle of the table, in days. If you execute the CREATE TABLE LIKE statement to create a table based on the schema of the source table, the lifecycle settings of the source table are not replicated.
- **CREATE TABLE [IF NOT EXISTS] table_name [AS select_statement | LIKE existing_table_name]:**
 - If you execute the CREATE TABLE...AS select_statement... statement to create a table, the data is replicated from the source table. However, the CREATE TABLE...AS select_statement... statement does not replicate the partition attributes of the source table. Partition key columns in the source table become standard columns in the destination table.
 - If you execute the CREATE TABLE...LIKE existing_table_name statement to create a table, the destination table has the same schema as the source table. However, the statement does not replicate data or lifecycle settings from the source table to the destination table.
- **Clauses related to external tables:**
 - **STORED BY StorageHandler:** the storage handler that is specified based on the format of data in the external table.
 - **WITH SERDEPROPERTIES (Options):** the parameters for authorization, compression, and character parsing in the external table.
 - **LOCATION OSSLocation:** the Object Storage Service (OSS) directory where the external table is saved.
- **TBLPROPERTY("compressionstrategy"="xxx"):** the attribute of the transactional table. You can specify the lifecycle and comment attributes in this clause. compressionstrategy is the compression policy for data storage. The valid values are normal, high, and extreme. The compressionstrategy parameter must be in lowercase. However, the value of this parameter is not case-sensitive. This attribute applies only to the internal tables of MaxCompute.

The default compression policy is normal for non-archived data and high for archived data.

 - **Definitions of the valid values:**
 - normal: indicates zstd level 1.
 - high: indicates zstd level 4.
 - extreme: indicates zstd level 9.
 - You can also specify the compression policy for a project. The project owner must use odps.storage.compression.strategy to specify the compression policy in the common attributes for project management. After this attribute is specified, the configuration applies to all generated non-archived data and does not affect the archive policy.
- **TBLPROPERTIES("transactional"="true"):** indicates that the table to create is a transactional table. You can execute the UPDATE or DELETE statement to update or delete data by row.

 **Note** If you do not specify IF NOT EXISTS in a table creation statement and another table with the same name exists, an error is returned. If you specify IF NOT EXISTS in a table creation statement, a success message is returned. The message is returned regardless of whether a table with the same name exists. The message is returned even if the schema of the existing table is different from that of the table to create. In addition, the metadata of the existing table remains unchanged.

Examples for creating a partitioned table

The following example shows how to create a table named sale_detail to store sales records. The table has two partition key columns: sale_date and region. Syntax:

```
CREATE TABLE IF NOT EXISTS sale_detail
(
shop_name STRING,
customer_id STRING,
total_price DOUBLE
)
PARTITIONED BY (sale_date STRING,region STRING);
-- Create a partitioned table named sale_detail.
```

CREATE TABLE...AS... example

You can execute the `CREATE TABLE...AS select_statement...` statement to create a table for which data is replicated from the source table. Example:

```
CREATE TABLE sale_detail_ctas1 AS
SELECT * FROM sale_detail;
```

If the `sale_detail` table contains data, all data in the `sale_detail` table is replicated to the `sale_detail_ctas1` table after you execute the preceding statement.

Note The `sale_detail` table is a partitioned table. When you execute the `CREATE TABLE...AS select_statement...` statement to create `sale_detail_ctas1`, partition attributes are not replicated and partition key columns in the `sale_detail` table are considered standard columns in the `sale_detail_ctas1` table. `sale_detail_ctas1` is a non-partitioned table that has five columns.

If the `SELECT` clause in the `CREATE TABLE...AS select_statement...` statement uses constants as column values, we recommend that you specify the names of columns. Example:

```
CREATE TABLE sale_detail_ctas2
AS
SELECT shop_name, customer_id, total_price, '2013' AS sale_date, 'China' AS region
FROM sale_detail;
```

Example without column aliases specified:

```
CREATE TABLE sale_detail_ctas3
AS
SELECT shop_name, customer_id, total_price, '2013', 'China'
FROM sale_detail;
```

CREATE TABLE...LIKE... example

If you want the source and destination tables to have the same schema, you can execute the `CREATE TABLE...LIKE...` statement. Example:

```
CREATE TABLE sale_detail_like LIKE sale_detail;
```

The schema of `sale_detail_like` is exactly the same as that of `sale_detail`. The tables have the same attributes, such as column names, column comments, and table comments, except for the lifecycle. However, data in `sale_detail` is not replicated to `sale_detail_like`.

Create clustered tables

Create hash-clustered tables:

```
CREATE TABLE T1 (a STRING, b STRING, c BIGINT) CLUSTERED BY (c) SORTED BY (c) INTO 1024 BUCKETS; -- Create a hash-clustered non-partitioned table.  
CREATE TABLE T1 (a STRING, b STRING, c BIGINT) PARTITIONED BY (dt STRING) CLUSTERED BY (c) SORTED BY (c) INTO 1024 BUCKETS; -- Create a hash-clustered partitioned table.
```

Create range-clustered tables:

```
CREATE TABLE T2 (a STRING, b STRING, c BIGINT) RANGE CLUSTERED BY (c) SORTED BY (c) INTO 1024 BUCKETS; -- Create a range-clustered non-partitioned table.  
CREATE TABLE T2 (a STRING, b STRING, c BIGINT) PARTITIONED BY (dt STRING) CLUSTERED BY (c) SORTED BY (c); -- Create a range-clustered partitioned table. The number of buckets can be empty.
```

Create transactional tables

```
-- Create a non-partitioned transactional table.  
create table txn_table(id bigint) tblproperties("transactional"="true");  
-- Create a partitioned transactional table.  
create table if not exists txn_table_pt(id bigint) partitioned by(ds string) tblproperties ("transactional"="true");
```

Query table information

MaxCompute allows you to execute the DESC statement to query table information.

```
DESC <table_name>;  
-- Query table information.  
DESC EXTENDED <table_name>;  
-- Query information about an external table.  
-- You can also check whether the table is a transactional table. If you want to check whether the table is a transactional table, you must upgrade the MaxCompute client to 0.35.0.
```

Query the statements that are used to create tables

MaxCompute allows you to execute the SHOW CREATE TABLE statement to query the statements that are used to create tables.

```
SHOW CREATE TABLE <table_name>;
```

 **Note** You can use this statement to generate SQL DDL statements that are used to create tables. These DDL statements can be used to rebuild the table schema.

1.6.4.1.2. Delete a table

This topic describes how to use a DDL statement to delete a table.

Syntax:

```
DROP TABLE [IF EXISTS] table_name;
```

 **Note** If you do not specify IF EXISTS and the table does not exist, an error is returned. If this option is specified, a success message is returned regardless of whether the table exists.

Examples:

```
CREATE TABLE sale_detail_drop LIKE sale_detail;
DROP TABLE sale_detail_drop;
-- If the table exists, a success message is returned. If the table does not exist, an error is returned.
DROP TABLE IF EXISTS sale_detail_drop2;
-- A success message is returned regardless of whether the sale_detail_drop2 table exists.
```

1.6.4.1.3. Rename a table

This topic describes how to use a DDL statement to rename a table.

Syntax:

```
ALTER TABLE table_name RENAME TO new_table_name;
```

Note

- The RENAME operation changes only the name of a table. Data in the table remains unchanged.
- If a table that has the same name as the table specified by new_table_name exists, an error is returned.
- If the table specified by table_name does not exist, an error is returned.

Examples:

```
CREATE TABLE sale_detail_rename1 LIKE sale_detail;
ALTER TABLE sale_detail_rename1 RENAME TO sale_detail_rename2;
```

1.6.4.1.4. Change the owner of a table

This topic describes how to use a DDL statement to change the owner of a table.

MaxCompute SQL allows you to execute the `CHANGEOWNER` statement to change the owner of a table.

Syntax:

```
ALTER TABLE table_name CHANGEOWNER TO 'ALIYUN$xxx@aliyun.com';
```

1.6.4.1.5. Modify the comment of a table

This topic describes how to use a DDL statement to modify the comment of a table.

Syntax:

```
ALTER TABLE table_name SET COMMENT 'tbl comment';
```

Note

- The table specified by table_name must exist.
- The value of comment can be up to 1,024 bytes.

Examples:

```
ALTER TABLE sale_detail SET COMMENT 'new coments for table sale_detail';
```

You can execute the DESC statement in MaxCompute to view the comment of the table after modification.

1.6.4.1.6. Modify the lifecycle settings of a table

MaxCompute provides the lifecycle management feature to facilitate storage release and simplify data reclamation. This topic describes how to use a DDL statement to modify the lifecycle settings of a table.

Syntax:

```
ALTER TABLE table_name SET lifecycle days;
```

Note

- days: the lifecycle duration, which must be a positive integer in days.
- table_name: the name of the table whose lifecycle settings you want to modify.
- You can specify the lifecycle when you create a table. The lifecycle can be specified only for a table rather than a partition. After a lifecycle is specified for a partitioned table, the lifecycle applies to the partitions of the table.
- If data in a table is modified, the value of LastDataModifiedTime is updated. Therefore, MaxCompute determines whether to reclaim this table based on the value of LastDataModifiedTime and the lifecycle settings.
- For a non-partitioned table, if the data remains unchanged for the duration specified by days after data is last modified, MaxCompute automatically executes a statement such as DROP TABLE to reclaim the table.
- For a partitioned table, MaxCompute determines whether to reclaim a partition based on the value of LastDataModifiedTime for the partition. Unlike non-partitioned tables, a partitioned table is not deleted even if all of the partitions are reclaimed.
- You can only modify the lifecycle settings instead of disabling the lifecycle for non-partitioned tables. You can disable the lifecycle for a specific partition in a partitioned table.

Examples:

```
create table test_lifecycle(key string) lifecycle 100;  
-- Create a table named test_lifecycle with a lifecycle of 100 days.  
alter table test_lifecycle set lifecycle 50;  
-- Change the lifecycle of the test_lifecycle table to 50 days.
```

1.6.4.1.7. Disable or enable the lifecycle feature

In some cases, if you do not want some partitions to be automatically reclaimed based on the lifecycle feature, you can disable the lifecycle feature for these partitions. This topic describes how to use a DDL statement to disable or enable the lifecycle feature.

Syntax:

```
ALTER TABLE table_name partition[partition_spec] ENABLE|DISABLE LIFECYCLE;
```

Note

- **DISABLE LIFECYCLE:** disables the lifecycle feature for a table or partition.
 - It prevents the reclamation of a table and its partitions based on the lifecycle feature. This parameter has a higher priority than `partition_spec enable lifecycle`. If you use `table disable lifecycle`, `partition_spec enable lifecycle` is invalid.
 - After the lifecycle feature of a table is disabled, the table lifecycle settings and the **ENABLE** and **DISABLE** tags of partitions in the table are retained.
 - After the lifecycle feature of a table is disabled, you can still modify the lifecycle settings of the table and its partitions.
- **ENABLE LIFECYCLE:** enables the lifecycle feature for a table or partition.
 - A table and its partitions can be reclaimed based on the lifecycle feature again. The lifecycle settings of the current table and its partitions are used by default.
 - Before you enable the lifecycle feature for a table, you can modify the lifecycle settings of the table and its partitions. This prevents data from being mistakenly reclaimed due to the use of previous settings.

Examples:

```
ALTER TABLE trans DISABLE LIFECYCLE;
-- Disable the lifecycle feature for the trans table.
ALTER TABLE trans PARTITION(dt='20141111') DISABLE LIFECYCLE;
-- Disable the lifecycle feature for the dt='20141111' partition in the trans table.
ALTER TABLE trans ENABLE LIFECYCLE;
-- Enable the lifecycle feature for the trans table.
ALTER TABLE trans PARTITION(dt='20141111') ENABLE LIFECYCLE;
-- Enable the lifecycle feature for the dt='20141111' partition in the trans table.
```

1.6.4.1.8. Modify the value of LastDataModifiedTime for a table

MaxCompute SQL allows you to perform the **TOUCH** operation to modify the value of `LastDataModifiedTime` for a table. This operation changes the value of `LastDataModifiedTime` for a table to the current time. This topic describes how to use a DDL statement to modify the value of `LastDataModifiedTime` for a table.

Syntax:

```
ALTER TABLE table_name TOUCH;
```

Note

- If the table specified by `table_name` does not exist, an error is returned.
- This operation modifies the value of `LastDataModifiedTime` for a table. In this case, MaxCompute considers a change to the table data and recalculates the lifecycle.

1.6.4.1.9. Modify the clustering attributes of a table

This topic describes how to use a DDL statement to modify the clustering attributes of a table.

MaxCompute allows you to add or remove the clustering attributes of a partitioned table by using the `ALTER TABLE` statement.

 Note

- The `ALTER TABLE` statement can modify only the clustering attributes of a partitioned table. The clustering attributes of a non-partitioned table cannot be modified after these attributes are added. The `ALTER TABLE` statement is applicable to the tables that already exist. After new clustering attributes are added, new partitions are stored based on the modified clustering attributes.
- The `ALTER TABLE` statement affects only the new partitions of partitioned tables, including the partitions generated by using the `INSERT OVERWRITE` statement. New partitions are stored based on the new clustering attributes, while the clustering attributes and storage of existing partitions remain unchanged. After you specify the clustering attributes for a table, you can remove the clustering attributes and add clustering attributes for the table again. You can specify different clustering columns, sort columns, and numbers of buckets for new partitions.
- The `ALTER TABLE` statement takes effect only on the new partitions of a table. Therefore, you cannot specify a partition in this statement.

Add hash clustering attributes to a table

Syntax:

```
ALTER TABLE table_name  
[CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...])] INTO number_of_buckets BUCKETS]
```

Remove hash clustering attributes from a table

Syntax:

```
ALTER TABLE table_name NOT CLUSTERED;
```

Add range clustering attributes to a table

If you do not specify the number of buckets, MaxCompute automatically determines the optimal number based on the data volume.

Syntax:

```
ALTER TABLE table_name  
[RANGE CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...])] INTO number_of_buckets BUCKETS]
```

Remove range clustering attributes from a table or partition

Syntax:

```
ALTER TABLE table_name NOT CLUSTERED;  
ALTER TABLE table_name partition_spec NOT CLUSTERED;
```

1.6.4.1.10. Delete data from a non-partitioned table

This topic describes how to use a DDL statement to delete data from a non-partitioned table.

Syntax:

```
TRUNCATE TABLE table_name;
```

Note This statement is used to delete data from a specific non-partitioned table. To delete data from a partitioned table, execute the `ALTER TABLE table_name DROP PARTITION(partition_spec)` statement.

1.6.4.1.11. Archive table data

This topic describes how to use a DDL statement to archive the data of a table.

If a project does not have enough space and data needs to be compressed or deleted, you can use table archiving of MaxCompute to compress data by about 50%. Table archiving uses a compression algorithm with a high compression ratio. It saves data as redundant array of independent disks (RAID) files. Data is no longer simply stored in three copies. Instead, six copies and three check blocks are maintained to increase the compression ratio from 1:3 to 1:1.5. Table archiving allows data to consume only half of the physical space.

However, this feature comes at a price. If a data block or machine is faulty, it requires a long time to restore the data block, and read performance deteriorates. Therefore, this feature is suitable for the compression of cold data. For example, you can use this feature to store a large number of logs that are less used as RAID files for a long time.

Syntax:

```
ALTER TABLE [table_name] <PARTITION(partition_name='partition_value')> ARCHIVE;
```

Note

- If the table specified by table_name does not exist, an error is returned.
- If the partition specified by partition_name does not exist, an error is returned.

Examples:

```
alter table my_log partition(ds='20170101') archive;
```

Output:

```
Summary:
table name: test0128 /pt=a instance count: 1 run time: 21
before merge, file count: 1 file size: 456 file physical size: 1368
after merge, file count: 1 file size: 512 file physical size: 768
```

Note

The output shows the changes in the logical size and physical size after the data is archived. In the archiving process, multiple small files are automatically merged. After the data is archived, you can execute the `DESC EXTENDED` statement to check whether the data in the partition has been archived and view the usage of physical space:

```
desc extended my_log partition(ds='20170101');
-- The following result is returned:
+-----+
PartitionSize: 512 |
+-----+
CreateTime: 2017-01-28 07:05:20 |
LastDDLTime: 2017-01-28 07:05:20 |
LastModifiedTime: 2017-01-28 07:05:21 |
+-----+
```

1.6.4.1.12. Forcibly delete data from a table or partition

This topic describes how to use a DDL statement to forcibly delete data from a table or partition.

If you want to forcibly and irrecoverably delete data from a table or partition to immediately release storage, you can execute a DROP statement with the PURGE option.

Syntax:

```
DROP TABLE tblname PURGE;
ALTER TABLE tblname DROP PARTITION(part_spec) PURGE;
```

Examples:

```
drop table my_log purge;
alter table my_log drop partition (ds='20170618') purge;
```

1.6.4.2. View-based operation

1.6.4.2.1. Create a view

This topic describes how to use a DDL statement to create a view.

Syntax:

```
CREATE [OR REPLACE] VIEW [IF NOT EXISTS] view_name
[(col_name [COMMENT col_comment], ...)]
[COMMENT view_comment]
[AS select_statement];
```

Note

- To create a view, you must have read permissions on the table referenced by the view.
- A view can contain only one valid SELECT statement.
- A view can reference other views but cannot reference itself. Circular references are not supported.
- You cannot write data to a view. For example, the `INSERT INTO` and `INSERT OVERWRITE` statements do not work on views.
- If the table referenced by a view changes, you may no longer be able to access the view. For example, a view becomes inaccessible after the table it references is deleted. You must maintain the mappings between referenced tables and views.
- If the `CREATE VIEW` statement is executed without the `IF NOT EXISTS` option and the view already exists, an exception is returned. In this case, you can execute `CREATE VIEW` or `REPLACE VIEW` to recreate the view. The permissions on the view remain unchanged after the view is recreated.

Examples:

```
create view if not exists sale_detail_view
(store_name, customer_id, price, sale_date, region)
comment 'a view for table sale_detail'
as select * from sale_detail;
-- Create a view named sale_detail_view.
```

1.6.4.2.2. Delete a view

This topic describes how to use a DDL statement to delete a view.

Syntax:

```
DROP VIEW [IF EXISTS] view_name;
```

Note If the partition does not exist and `IF EXISTS` is not specified, an error is returned.

Examples:

```
DROP VIEW IF EXISTS sale_detail_view;
```

1.6.4.2.3. Rename a view

This topic describes how to use a DDL statement to rename a view.

Syntax:

```
ALTER VIEW view_name RENAME TO new_view_name;
```

Note If a view with the same name already exists, an error is returned.

Examples:

```
alter view sale_detail_view rename to market;
```

1.6.4.3. Column and partition operations

1.6.4.3.1. Add partitions

This topic describes how to use a DDL statement to add partitions.

Syntax:

```
alter table table_name add [if not exists] partition partition_spec;  
-- Add one partition at a time.  
alter table table_name add [if not exists] partition partition_spec [PARTITION partition_spec PARTITION partition_spec...];  
-- Add multiple partitions at a time.
```

Note

- If you do not specify IF NOT EXISTS and another partition with the same name exists, an error is returned.
- A MaxCompute table can contain a maximum of 60,000 partitions.
- To add a partition to a table that has multi-level partitions, you must specify all partition values.

Examples:

```
alter table sale_detail add if not exists partition (sale_date='201712', region='hangzhou');  
-- Add a partition to store the sales records of the China (Hangzhou) region for December 2017.  
alter table sale_detail add if not exists partition (sale_date='201712', region='shanghai');  
-- Add a partition to store the sales records of the China (Shanghai) region for December 2017.  
alter table sale_detail add if not exists partition(sale_date='20171011');  
-- Specify only the sale_date partition. An error is returned.  
alter table sale_detail add if not exists partition(region='shanghai');  
-- Specify only the region partition. An error is returned.  
alter table sale_detail add if not exists partition (sale_date='201712', region='hangzhou') partition (sale_date='201712', region='shanghai');  
-- Add two partitions to store the sales records of the China (Hangzhou) and China (Shanghai) regions for December 2017.
```

1.6.4.3.2. Delete a partition

This topic describes how to use DDL statements to delete a partition.

MaxCompute allows you to delete partitions that meet a specific filter condition. If you want to delete one or more partitions that meet a filter condition at a time, you can use an expression to specify the condition, use the condition to filter partitions, and then delete the partitions.

If you do not specify a filter condition, execute the following statements:

```
alter table table_name drop [if exists] PARTITION partition_spec;  
-- Delete one partition at a time.  
alter table table_name drop [if exists] PARTITION partition_spec,PARTITION partition_spec,[PARTITION partition_spec...];  
-- Delete multiple partitions at a time.
```

 **Note** Syntax for `partition_spec`:

```
partition_spec:(partition_col1 = partition_col_value1, partition_col2 = partition_col_value2, ...)
```

If you specify a filter condition, execute the following statement:

```
ALTER TABLE table_name DROP [IF EXISTS] PARTITION partition_filtercondition;
-- Delete a partition based on the specified filter condition.
```

 **Note** Syntax for `PARTITION partition_filtercondition`:

```
PARTITION partition_filtercondition
: PARTITION (partition_col relational_operators partition_col_value)
| PARTITION (scalar(partition_col) relational_operators partition_col_value)
| PARTITION (partition_filtercondition1 AND|OR partition_filtercondition2)
| PARTITION (NOT partition_filtercondition)
| PARTITION (partition_filtercondition1)[,PARTITION (partition_filtercondition2), ...]
```

Parameters:

- `table_name`: the name of the partitioned table from which you want to delete a partition.
- `IF EXISTS`: If `IF EXISTS` is not specified and the partition does not exist, an error is returned.
- `partition_spec`: the name of the partition that you want to delete. The value is not case-sensitive. `partition_col` indicates the name of the partition key column, and `partition_col_value` indicates the value of the partition key column.
- `PARTITION partition_filtercondition`: the partition that you want to delete. The value is not case-sensitive.
 - `partition_col`: the name of the partition key column.
 - `relational_operators`: the relational operator.
 - `partition_col_value`: a value in the partition key column, which is a comparison value or regular expression. The data type of this value must be the same as that of the partition key column.
 - `scalar()`: the scalar function. This function generates a scalar based on the input value, processes the value in `partition_col`, and uses the relational operator specified by `relational_operators` to compare the processed value with `partition_col_value`.
 - The filter condition that is used to delete partitions supports the following logical operators: NOT, AND, and OR. You can use `PARTITION (NOT partition_filtercondition)` to obtain the complementary set of the condition. You can use `PARTITION (partition_filtercondition1 AND|OR partition_filtercondition2)` to obtain the condition that is used to match the partitions you want to delete.
 - Multiple `PARTITION partition_filtercondition` clauses are supported. If these clauses are separated by commas (,), OR is used for each clause to obtain the condition that is used to match the partitions you want to delete.

Limits:

- Each `PARTITION partition_filtercondition` clause can be used for only one partition key column.
- If you use an expression to specify `PARTITION partition_filtercondition`, only the built-in scalar function can be used for the expression.

Examples:

The filter condition is not specified.

```
-- Delete a partition from the sale_detail table. The partition stores the sales records of the China (Hangzhou) region in December 2017.
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date='201712',region='hangzhou');
-- Delete two partitions from the sale_detail table at a time. The partitions store the sales records of the China (Hangzhou) and China (Shanghai) regions in December 2017.
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date='201712',region='hangzhou'),PARTITION(sale_date='201712',region='shanghai');
```

The filter condition is specified.

```
-- Create a partitioned table.
CREATE TABLE IF NOT EXISTS sale_detail(
shop_name STRING,
customer_id STRING,
total_price DOUBLE)
PARTITIONED BY (sale_date STRING);
-- Add partitions.
ALTER TABLE sale_detail ADD IF NOT EXISTS
PARTITION (sale_date= '201910')
PARTITION (sale_date= '201911')
PARTITION (sale_date= '201912')
PARTITION (sale_date= '202001')
PARTITION (sale_date= '202002')
PARTITION (sale_date= '202003')
PARTITION (sale_date= '202004')
PARTITION (sale_date= '202005')
PARTITION (sale_date= '202006')
PARTITION (sale_date= '202007');
-- Delete multiple partitions from the table at a time.
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date < '201911');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date >= '202007');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date LIKE '20191%');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date IN ('202002','202004','202006'));
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date BETWEEN '202001' AND '202007');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(substr(sale_date, 1, 4) = '2020');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date < '201912' OR sale_date >= '202006');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date > '201912' AND sale_date <= '202004');
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(NOT sale_date > '202004');
-- Delete partitions by using a condition. The condition is specified by expressions that have the OR relationship.
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date < '201911'), PARTITION(sale_date >= '202007');
-- Add partitions in other formats.
ALTER TABLE sale_detail ADD IF NOT EXISTS
PARTITION (sale_date= '2019-10-05');
PARTITION (sale_date= '2019-10-06')
PARTITION (sale_date= '2019-10-07');
-- Delete multiple partitions at a time and use regular expressions to match the partitions you want to delete.
ALTER TABLE sale_detail DROP IF EXISTS PARTITION(sale_date RLIKE '2019-\\d+\\d+');
-- Create a table that has multi-level partitions.
CREATE TABLE IF NOT EXISTS region_sale_detail(
shop_name STRING,
customer_id STRING,
total_price DOUBLE)
PARTITIONED BY (sale_date STRING , region STRING );
-- Add partitions.
ALTER TABLE region_sale_detail ADD IF NOT EXISTS
PARTITION (sale_date= '201910',region = 'shanghai')
PARTITION (sale_date= '201911',region = 'shanghai')
```

```

PARTITION (sale_date= '201911',region = 'shanghai')
PARTITION (sale_date= '201912',region = 'shanghai')
PARTITION (sale_date= '202001',region = 'shanghai')
PARTITION (sale_date= '202002',region = 'shanghai')
PARTITION (sale_date= '201910',region = 'beijing')
PARTITION (sale_date= '201911',region = 'beijing')
PARTITION (sale_date= '201912',region = 'beijing')
PARTITION (sale_date= '202001',region = 'beijing')
PARTITION (sale_date= '202002',region = 'beijing');
-- Delete the partitions that meet the sale_date < '201911' and region = 'beijing' conditions from the table at a time.
ALTER TABLE region_sale_detail DROP IF EXISTS PARTITION(sale_date < '201911'),PARTITION(region = 'beijing');

```

If you execute the following statement, the following error is returned: **FAILED: ODPS-0130071:[1.82] Semantic analysis exception - invalid column reference region, partition expression must have one and only one column reference**.

```

-- An error is returned because PARTITION partition_filtercondition can be specified only for one partition key column.
ALTER TABLE region_sale_detail DROP IF EXISTS PARTITION(sale_date < '201911' AND region = 'beijing');

```

1.6.4.3.3. Add columns or comments

This topic describes how to use DDL statements to add columns or comments.

Syntax:

```

ALTER TABLE table_name ADD COLUMNS (col_name1 type1,col_name2 type2...);
-- Add columns.
ALTER TABLE table_name ADD COLUMNS (col_name1 type1 comment 'XXX',col_name2 type2 comment 'XXX');
-- Add both columns and comments.

```

Examples:

```

ALTER TABLE sale_detail ADD COLUMNS (customer_name STRING, education BIGINT);
-- Add two columns to the sale_detail table.
ALTER TABLE sale_detail ADD COLUMNS (customer_name STRING comment 'Customer', education BIGINT comment 'Education');
-- Add two columns and their comments to the sale_detail table.

```

1.6.4.3.4. Change the name of a column

This topic describes how to use a DDL statement to change the name of a column.

Syntax:

```

ALTER TABLE table_name CHANGE COLUMN old_col_name RENAME TO new_col_name;

```

Note

- The column specified by `old_col_name` must exist in the table.
- The table cannot contain the column specified by `new_col_name`.

Examples:

```
ALTER TABLE sale_detail CHANGE COLUMN customer_name RENAME TO customer;  
-- Change the name of a column in the sale_detail table.
```

1.6.4.3.5. Modify the comment of a column

This topic describes how to use a DDL statement to modify the comment of a column.

Syntax:

```
ALTER TABLE table_name CHANGE COLUMN col_name COMMENT 'comment_string';
```

Note

- The value of `comment_string` cannot exceed 1,024 bytes.
- The data type and position of a column cannot be changed.
- The column specified by `col_name` must exist.

Examples:

```
ALTER TABLE sale_detail CHANGE COLUMN customer COMMENT 'customer';  
-- Modify the comment of a column in the sale_detail table.
```

1.6.4.3.6. Modify the name and comment of a column at the same time

This topic describes how to use a DDL statement to modify the name and comment of a column at the same time.

Syntax:

```
ALTER TABLE table_name CHANGE COLUMN old_col_name new_col_name column_type COMMENT 'column_comment';
```

Note

- The column specified by `old_col_name` must exist.
- The table cannot contain the column specified by `new_col_name`.
- The data type and position of a column cannot be changed.
- The value of `column_comment` cannot exceed 1,024 bytes.

Examples:

```
ALTER TABLE sale_detail CHANGE COLUMN customer customer_name string COMMENT 'Customer';  
-- Modify the name and comment of a column in the sale_detail table.
```

1.6.4.3.7. Modify the value of LastDataModifiedTime for a non-partitioned table or a partition in a partitioned table

MaxCompute SQL allows you to execute the TOUCH operation to modify the value of LastDataModifiedTime for a non-partitioned table or a partition in a partitioned table. This operation changes the value of LastDataModifiedTime to the current time. This topic describes how to use a DDL statement to modify the value of LastDataModifiedTime for a non-partitioned table or a partition in a partitioned table.

Syntax:

```
ALTER TABLE table_name TOUCH;
-- Modify the value of LastDataModifiedTime for a non-partitioned table.
ALTER TABLE table_name TOUCH PARTITION(partition_col='partition_col_value', ...);
-- Modify the value of LastDataModifiedTime for a partition in a partitioned table.
```

Note

- If the table specified by table_name does not exist, an error is returned.
- If the partition specified by partition_col='partition_col_value' does not exist, an error is returned.
- This operation modifies the value of LastDataModifiedTime for a non-partitioned table or a partition in a partitioned table. In this case, MaxCompute considers a change to the table data and recalculates the lifecycle.

Examples:

```
ALTER TABLE result_table TOUCH;
-- Modify the value of LastDataModifiedTime for a non-partitioned table.
ALTER TABLE sale_detail TOUCH PARTITION(sale_date='201712');
-- Modify the value of LastDataModifiedTime for a partition in a partitioned table.
```

1.6.4.3.8. Modify partition values

This topic describes how to use a DDL statement to modify partition values.

Syntax:

```
ALTER TABLE table_name PARTITION (partition_col1 = 'partition_col_value1', partition_col2 = 'partition_col_value2', ...)
RENAME TO PARTITION (partition_col1 = 'partition_col_newvalue1', partition_col2 = 'partition_col_newvalue2', ...);
```

Note

- This statement can modify only the values rather than the names of partition key columns.
- To modify the values of one or more partitions in a table that has multi-level partitions, you must specify the partition values at each level.
- If the table specified by table_name does not exist, an error is returned.

Examples:

```
ALTER TABLE sale_detail PARTITION (sale_date = '201712', region = 'hangzhou') RENAME TO PARTITION (sale_date = '201710', region = 'beijing');  
-- Modify the partition values in the sale_detail table.
```

1.6.4.3.9. Merge partitions

MaxCompute SQL allows you to execute the `MERGE PARTITION` statement to merge multiple partitions of a partitioned table into one partition. This operation deletes the dimension information about the merged partitions and transfers data to a specific partition. This topic describes how to use a DDL statement to merge partitions.

Syntax:

```
ALTER TABLE <tableName> MERGE [IF EXISTS] PARTITION(<predicate>) [, PARTITION(<predicate2>) ...] OVERWRITE PARTITION(<fullPartitionSpec>) [PURGE];
```

Note

- If you do not specify `IF EXISTS` and the partition you want to merge does not exist, an error is returned.
- If you specify `IF EXISTS` but no partitions meet the merge conditions, no new partitions are generated.
- If source data is concurrently modified by operations such as `INSERT`, `RENAME`, and `DROP` when you execute the preceding statement, an error is returned even though you have specified `IF EXISTS`.
- If the `PURGE` attribute is specified, merged partitions cannot be restored by using `Kunlunjing`.

Limits and troubleshooting

- Extreme storage is not supported.
- Tables that depend on the file order are not supported, such as tables in `Xlib` or `ALGO`.
- External tables and table shards are not supported. After the partitions of a clustered table are merged, the merged partitions do not have the clustering attributes.
- Hash operations are performed by a catalog server on tables to merge partitions. A capacity limit is imposed on merged partitions. A hard link in Apsara Distributed File System can have a maximum of seven copies.
- You can merge a maximum of 4,000 partitions at a time.
- The number of partitions that can wait on a catalog server for merging is 10 million.
- If an error that indicates the catalog server is busy is returned, try again later.
- If a hard link in Apsara Distributed File System is faulty, clear the recycle bin and try again.

Examples:

```

odps@ jet_zwz>SHOW PARTITIONS intpstringstringstring;
ds=20181101/hh=00/mm=00
ds=20181101/hh=00/mm=10
ds=20181101/hh=10/mm=00
ds=20181101/hh=10/mm=10
OK
odps@ jet_zwz>DESC intpstringstringstring;
+-----+-----+-----+-----+
| value | ds   | hh   | mm   |
+-----+-----+-----+-----+
| 1     | 20181101 | 00   | 00   |
| 1     | 20181101 | 00   | 10   |
| 1     | 20181101 | 10   | 00   |
| 1     | 20181101 | 10   | 10   |
+-----+-----+-----+-----+
-- Partitions and data of a partitioned table:
odps@ jet_zwz>ALTER TABLE intpstringstringstring MERGE PARTITION(hh='00') OVERWRITE PARTITION(ds='20181101', hh='00', mm='00');
ID = 20190404025755844g80qwa7a
OK
-- Merge all partitions that meet the hh='00' condition into the ds=20181101/hh=00/mm=00 partition.
odps@ jet_zwz>SHOW PARTITIONS intpstringstringstring;
ds=20181101/hh=00/mm=00
ds=20181101/hh=10/mm=00
ds=20181101/hh=10/mm=10
OK
-- Query the partitions of the table after partition merge.
odps@ jet_zwz>DESC intpstringstringstring;
+-----+-----+-----+-----+
| value | ds   | hh   | mm   |
+-----+-----+-----+-----+
| 1     | 20181101 | 00   | 00   |
| 1     | 20181101 | 00   | 00   |
| 1     | 20181101 | 10   | 00   |
| 1     | 20181101 | 10   | 10   |
+-----+-----+-----+-----+
-- Data in two partitions that meet the specified condition is merged into the specified partition.

```

1.6.5. DML statements

1.6.5.1. INSERT

1.6.5.1.1. Update data in a table

This topic describes how to use an INSERT statement to update data in a table.

The `INSERT OVERWRITE` and `INSERT INTO` statements are used to write the data processing results of MaxCompute SQL to a destination table.

The `INSERT INTO` statement inserts data into a table or partition. The `INSERT INTO` statement cannot be used to insert data into clustered tables. If you want to insert a small amount of test data, you can use this statement with `VALUES`.

The `INSERT OVERWRITE` statement clears the original data in a table or partition and inserts data into the table or partition. The `INSERT OVERWRITE` statement cannot be used to insert data into specific columns.

Syntax:

```
INSERT OVERWRITE|INTO TABLE table_name [PARTITION (partcol1=val1, partcol2=val2 ...)] [(col1,col2 ...)]
select_statement
FROM from_statement
[ZORDER BY zcol1 [, zcol2 ...]];
```

Note

- The INSERT syntax in MaxCompute is different from that in MySQL or Oracle. In MaxCompute, `INSERT OVERWRITE` or `INSERT INTO` must be followed by the keyword `TABLE`, instead of directly followed by the `table_name` parameter.
- To ensure data consistency during concurrent writes, MaxCompute uses the atomicity, consistency, isolation, durability (ACID) properties.
- The mappings between the source and destination tables are based on the column sequence in the `SELECT` clause. The mappings between the column names of the tables are not considered.
- If the destination table consists of static partitions and you want to insert data into a static partition, partition key columns cannot be included in the `SELECT` clause.

Parameters:

- `table_name`: the name of the table into which you want to insert data.
- `PARTITION (partcol1=val1, partcol2=val2 ...)`: the name of the partition into which you want to insert data. The value must be a constant. It cannot be an expression, such as a function.
- `[(col1,col2 ...)]`: the names of the columns into which you want to insert data. This parameter is not supported in the `INSERT OVERWRITE` statement.
- `select_statement`: the `SELECT` clause used to query the data that you want to insert from the source table.
- `from_statement`: the `FROM` clause used to indicate the data source. For example, the value can be the name of the source table.
- `[ZORDER BY zcol1 [, zcol2 ...]]`: the columns that are used to sort data. If you write data to a table or partition, you can use this clause to place rows with similar data records in the columns specified in `select_statement` in adjacent positions. This improves filtering performance for queries and reduces storage costs.

The `ORDER BY x,y` clause sorts data records based on the ordering of `x` coming before `y`. The `ZORDER BY x,y` clause places rows that have similar `x` values in adjacent positions and rows that have similar `y` values in adjacent positions. If the filter condition of an SQL query statement includes sort columns, the `ORDER BY` clause filters and sorts data based on `x`, whereas the `ZORDER BY` clause filters and sorts data based on `x` or on both `x` and `y`. `ZORDER BY` increases the column store ratio.

Note

- The mappings between the source and destination tables are based on the column sequence in the `SELECT` clause. The mappings between the column names of the tables are not considered.
- If the destination table consists of static partitions and you want to insert data into a partition, partition key columns cannot be included in the `SELECT` clause.
- The `ZORDER BY` clause occupies a large number of resources to write data. As a result, data writes based on `ZORDER BY` require a longer time than data writes without ordering.
- If the destination table is a clustered table, `ZORDER BY` is not supported.
- `ZORDER BY` can be used with `DISTRIBUTE BY`. However, `ZORDER BY` cannot be used with `ORDER BY`, `CLUSTER BY`, or `SORT BY`.

Examples:

Calculate the sales of different regions listed in the sale_detail table. Then, insert the obtained data into the sale_detail_insert table.

```
-- Create a partitioned table named sale_detail, add partitions, and insert data into the table. You do not need to define partition key columns as standard columns in the statement that is used to create the source table.
CREATE TABLE IF NOT EXISTS sale_detail
(
shop_name string,
customer_id string,
total_price double
)
PARTITIONED BY (sale_date STRING,region STRING);
-- Add partitions to the source table.
ALTER TABLE sale_detail ADD PARTITION (sale_date='2013', region='china');
-- Insert data into the source table. INSERT INTO TABLE table_name can be abbreviated as INSERT INTO table_name. However, INSERT OVERWRITE TABLE table_name has no abbreviation.
INSERT INTO sale_detail PARTITION (sale_date='2013', region='china') VALUES ('s1','c1',100.1),('s2','c2',100.2),('s3','c3',100.3);
-- Create a destination table named sale_detail_insert that has the same schema as the source table.
CREATE TABLE sale_detail_insert LIKE sale_detail;
-- Add partitions to the destination table.
ALTER TABLE sale_detail_insert ADD PARTITION (sale_date='2013', region='china');
-- Extract data from the sale_detail table and insert the data into the sale_detail_insert table.
-- Fields in the destination table do not need to be declared or rearranged.
-- If the destination table consists of static partitions and partition fields have been declared in PARTITION(), you do not need to include these fields in the SELECT clause. You only need to search for the fields based on the sequence of standard columns in the destination table and sequentially map these fields to those in the destination table. If the destination table consists of dynamic partitions, partition fields must be included in the SELECT clause. For more information, see Insert data into dynamic partitions.
INSERT OVERWRITE TABLE sale_detail_insert PARTITION (sale_date='2013', region='china')
SELECT
shop_name,
customer_id,
total_price
FROM sale_detail
ORDER BY customer_id, total_price;
```

Take note of the following points:

- The mappings between the source and destination tables are based on the column sequence in the SELECT clause. The mappings between the column names of the tables are not considered. Example:

```
INSERT OVERWRITE TABLE sale_detail_insert PARTITION (sale_date='2013', region='china')
SELECT customer_id, shop_name, total_price FROM sale_detail;
```

In the created sale_detail_insert table, the columns shop_name STRING, customer_id STRING, and total_price BIGINT are listed in sequence. If you insert data from the sale_detail table to the sale_detail_insert table, the data is inserted from the customer_id, shop_name, and total_price columns in sequence. In this case, data in sale_detail.customer_id is inserted into sale_detail_insert.shop_name, and data in sale_detail.shop_name is inserted into sale_detail_insert.customer_id.

- If you insert data into a partition, partition key columns cannot be included in the SELECT clause. If you execute the following statement, an error is returned because sale_date and region are partition key columns. These columns cannot be included in the INSERT statement that is used to insert table data into a static partition.

```
INSERT OVERWRITE TABLE sale_detail_insert PARTITION (sale_date='2013', region='china')
SELECT shop_name, customer_id, total_price, sale_date, region FROM sale_detail;
```

- The value of PARTITION must be a constant and cannot be an expression. The following example shows invalid settings:

```
INSERT OVERWRITE TABLE sale_detail_insert PARTITION (sale_date=datepart('2016-09-18 01:10:00', 'yyyy'), region='china')
SELECT shop_name, customer_id, total_price FROM sale_detail;
```

Precautions

If you want to update table data to a dynamic partition, take note of the following points:

- If you execute the INSERT INTO PARTITION statement and the specified partition does not exist, the system automatically creates this partition.
- If multiple jobs that use the INSERT INTO PARTITION statement concurrently run and the specified partitions do not exist, the system attempts to create these partitions. However, only one partition can be created.
- If you cannot control the concurrency of the jobs that use the INSERT INTO PARTITION statement, we recommend that you use the ALTER TABLE statement to create partitions in advance.

1.6.5.1.2. Insert data into multiple objects

This topic describes how to use an INSERT statement to insert data to multiple objects.

MaxCompute SQL allows you to insert data into different result tables or partitions by using one SQL statement.

Syntax:

```
FROM from_statement
INSERT OVERWRITE | INTO TABLE tablename1 [PARTITION (partcol1=val1, partcol2=val2 ...)]
select_statement1 [FROM from_statement]
[INSERT OVERWRITE | INTO TABLE tablename2 [PARTITION (partcol1=val3, partcol2=val4 ...)]
select_statement2 [FROM from_statement]];
```

Note

- In most cases, an SQL statement supports up to 256 outputs. If the number of outputs exceeds 256, a syntax error is returned.
- In a MULTI INSERT statement, non-partitioned tables and destination partitions in a partitioned table must be unique.
- The INSERT OVERWRITE and INSERT INTO operations cannot be simultaneously performed on different partitions of a table. If these operations are simultaneously performed on the partitions, an error is returned.
- PARTITION (partcol1=val1, partcol2=val2 ...): The parameter values can be only constants rather than expressions, such as functions.

Examples:

```

-- Create a destination table named sale_detail_multi.
create table sale_detail_multi like sale_detail;
-- Insert data from sale_detail into sale_detail_multi.
set odps.sql.allow.fullscan=true; -- Enable the full table scan, which is valid only for this session.
from sale_detail
insert overwrite table sale_detail_multi partition (sale_date='2010', region='china' )
select shop_name, customer_id, total_price
insert overwrite table sale_detail_multi partition (sale_date='2011', region='china' )
select shop_name, customer_id, total_price ;
-- An error is returned because the same partition appears more than once.
from sale_detail
insert overwrite table sale_detail_multi partition (sale_date='2010', region='china' )
select shop_name, customer_id, total_price
insert overwrite table sale_detail_multi partition (sale_date='2010', region='china' )
select shop_name, customer_id, total_price;
-- An error is returned because the INSERT OVERWRITE and INSERT INTO operations are simultaneously performed
on different partitions of a partitioned table.
from sale_detail
insert overwrite table sale_detail_multi partition (sale_date='2010', region='china' )
select shop_name, customer_id, total_price
insert into table sale_detail_multi partition (sale_date='2011', region='china' )
select shop_name, customer_id, total_price;

```

1.6.5.1.3. Insert data into dynamic partitions

This topic describes how to use an INSERT statement to insert data into dynamic partitions.

You can use the INSERT OVERWRITE statement to insert data into the following partitions of a partitioned table:

- Static partitions: You must specify partition values in the INSERT statement to insert data into the specified partitions.
- Dynamic partitions: You need only to specify the names of partition key columns in the INSERT statement. The values of the partition key columns are provided in the SELECT clause. The system automatically inserts data into the required partitions based on the values.

Syntax:

```

INSERT OVERWRITE|INTO TABLE tablename PARTITION (partcol1, partcol2 ...)
select_statement FROM from_statement;

```

 Note

- A maximum of 10,000 dynamic partitions can be generated by using the INSERT INTO statement. A maximum of 60,000 dynamic partitions can be generated by using the INSERT OVERWRITE statement.
- An SQL statement that is used to insert data into dynamic partitions in a distributed environment can generate a maximum of 512 dynamic partitions in a single process. If the number of dynamic partitions exceeds this limit, an exception occurs.
- The dynamically generated partition values cannot be NULL or contain special characters. If a value is NULL or contains special characters, an exception occurs.

```
FAILED: ODPS-0123031:Partition exception - invalid dynamic partition value: province=xxx
```

- If the destination table has multi-level partitions, you can specify some partitions as static partitions in an INSERT statement. However, the static partitions must be high-level partitions.
- If the destination table is a hash-clustered table, dynamic partitions are not supported.
- If you insert data into dynamic partitions, the mappings between the columns in select_statement and dynamic partitions in the destination table are determined by the column sequence, not by the column names. If the sequence of columns in the source table is different from that in the destination table, we recommend that you specify the columns in select_statement based on the column sequence in the destination table.

Examples:

Insert data from a source table into a destination table. You can obtain the partitions generated based on the region field only after the required statement is executed.

```
-- Create a destination table named total_revenues.  
create table total_revenues (revenue bigint) partitioned by (region string);  
-- Insert the data from sale_detail into total_revenues.  
insert overwrite table total_revenues partition(region)  
select total_price as revenue, region from sale_detail;
```

Insert data from a source table into a destination table. If the destination table has multi-level partitions, level-1 partition sale_date must be specified.

```
insert overwrite table sale_detail_dypart partition (sale_date='2013', region)  
select shop_name,customer_id,total_price,region from sale_detail;
```

To insert data into dynamic partitions, the dynamic partition key columns must be specified in the SELECT clause. Otherwise, the execution fails.

```
insert overwrite table sale_detail_dypart partition (sale_date='2013', region)  
select shop_name,customer_id,total_price from sale_detail;
```

If you specify only low-level subpartitions when you insert data into dynamic partitions, you may fail to insert data into high-level partitions.

```
insert overwrite table sales partition (region='china', sale_date)  
select shop_name,customer_id,total_price,sale_date from sale_detail;
```

If the data types of values in partition key columns are inconsistent with those in the SELECT clause, an error is returned when data is inserted into dynamic partitions in MaxCompute V1.0. MaxCompute V2.0 supports the implicit conversions of data types.

```
-- Create a destination table named parttable.
create table parttable(a int, b double) partitioned by (p string);
-- Insert the data from source table src into destination table parttable.
insert into parttable partition(p) select key, value, current_timestamp() from src;
-- Query data in parttable.
select * from parttable;
```

1.6.5.1.4. VALUES

This topic describes how to use the INSERT ...VALUES statement to insert data into a table that contains small amounts of data.

Syntax:

```
INSERT INTO TABLE tablename
[PARTITION (partcol1=val1, partcol2=val2,...)][(col1name1,colname2,...)]
[VALUES (col1_value,col2_value,...),(col1_value,col2_value,...),...]
```

Parameters:

- **tablename**: the name of the table into which you want to insert data. The table must be an existing table.
- **[PARTITION (partcol1=val1, partcol2=val2,...)]**: the information about partitions. If you want to insert data into a partitioned table, you must specify this parameter.
- **[(col1name1,colname2,...)]**: the names of the columns in the destination table.
- **col_value**: the value in the specified column in the destination table. Separate multiple values with commas (.). Each value in the column must be a constant. If no values are specified, the default value NULL is used.

Note

- You can use only the **INSERT INTO** statement rather than the **INSERT OVERWRITE** statement to insert data into specific columns.
- Functions cannot be used to construct constants for some complex data types, such as ARRAY, in VALUES. However, you can use the following statement to pass the values of the ARRAY type to VALUES:

```
INSERT INTO TABLE srcp (p='abc') SELECT 'a', ARRAY('1', '2', '3');
```

- To pass the values of the DATETIME or TIMESTAMP type, you must specify the data type in VALUES.

```
INSERT INTO TABLE srcp (p='abc') VALUES (datetime'2017-11-11 00:00:00',timestamp'2017-11-11 00:00:00.123456789');
```

Examples:

Insert data into a specific partition.

```
-- Delete the srcp table that already exists in the system.
DROP TABLE IF EXISTS srcp;
-- Create a partitioned table named srcp.
CREATE TABLE IF NOT EXISTS srcp (key string,value bigint) PARTITIONED BY (p string);
-- Insert data into the abc partition of the srcp table.
INSERT INTO TABLE srcp PARTITION (p='abc') VALUES ('a',1),('b',2),('c',3);
-- Query data from the srcp table.
SELECT * FROM srcp WHERE p='abc';
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| a    | 1     | abc  |
| b    | 2     | abc  |
| c    | 3     | abc  |
+-----+-----+-----+
```

Insert data into any partition.

```
-- Delete the srcp table that already exists in the system.
DROP TABLE IF EXISTS srcp;
-- Create a partitioned table named srcp.
CREATE TABLE IF NOT EXISTS srcp (key string,value bigint) PARTITIONED BY (p string);
-- Insert data into a partition of the srcp table.
INSERT INTO TABLE srcp PARTITION (p)(key,p) VALUES ('d','20170101'),('e','20170101'),('f','20170101');
-- Query data from the srcp table.
SELECT * FROM srcp WHERE p='20170101';
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| d    | NULL  | 20170101 |
| e    | NULL  | 20170101 |
| f    | NULL  | 20170101 |
+-----+-----+-----+
```

Use complex data types to construct constants and execute an `INSERT` statement to import data.

```
-- Create a partitioned table named srcp.
create table if not exists srcp (key string,value array<int>) partitioned by (p string);
-- Add a partition to the srcp table.
alter table srcp add if not exists partition (p='abc');
-- Insert data into the abc partition of the srcp table.
insert into table srcp partition (p='abc') select 'a', array(1, 2, 3);
-- Query data from the srcp table.
select * from srcp where p='abc';
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| a    | [1,2,3] | abc  |
+-----+-----+-----+
```

Use the `INSERT...VALUES` statement to write data of the DATETIME or TIMESTAMP type.

```
-- Create a partitioned table named srcp.
create table if not exists srcp (key string, value timestamp) partitioned by (p string);
-- Add a partition to the srcp table.
alter table srcp add if not exists partition (p='abc');
-- Insert data into the abc partition of the srcp table.
insert into table srcp partition (p='abc') values (datetime'2017-11-11 00:00:00',timestamp'2017-11-11 00:00:00.123456789');
-- Query data from the srcp table.
select * from srcp where p='abc';
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| 2017-11-11 00:00:00 | 2017-11-11 00:00:00.123 | abc  |
+-----+-----+-----+
```

Features of VALUES TABLE

When you use `INSERT...VALUES`, the values after `VALUES` must be constants. If you want to perform simple computing operations on the inserted data, we recommended that you use `VALUES TABLE` of MaxCompute.

Syntax:

```
values (<col1_value>,<col2_value>,...),(<col1_value>,<col2_value>,...),<table_name> (<col1_name> ,<col2_name>,... )...
```

You can use `VALUES TABLE` in the following scenarios:

- If no physical tables are available, create a table that contains multiple rows of data and perform computing operations on the table.

For example, `VALUES (...),(...) t(a,b)` defines that a table named `t` contains the `a` and `b` columns. The data type of the `a` column is `STRING` and that of the `b` column is `BIGINT`. The data type of a column must be derived from the `VALUES` list.

```
-- Delete the srcp table that already exists in the system.
DROP TABLE IF EXISTS srcp;
-- Create a partitioned table named srcp.
CREATE TABLE IF NOT EXISTS srcp (key string,value bigint) PARTITIONED BY (p string);
-- Insert data into the srcp table.
INSERT INTO TABLE srcp PARTITION (p) SELECT concat(a,b), length(a)+length(b),'20170102' FROM VALUES ('d',4),('e',5),('f',6) t(a,b);
-- Query data from the srcp table.
SELECT * FROM srcp WHERE p='20170102';
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| d4   | 2     | 20170102 |
| e5   | 2     | 20170102 |
| f6   | 2     | 20170102 |
+-----+-----+-----+
```

- Use VALUES TABLE rather than the combination of SELECT * FROM and UNION ALL to create a constant-type table.

```
SELECT 1 c UNION ALL SELECT 2 c;
-- The preceding statement is equivalent to the following statement:
SELECT * FROM VALUES (1), (2) t(c);
```

The following result is returned:

```
+-----+
| c    |
+-----+
| 1    |
| 2    |
+-----+
```

- Special forms of VALUES TABLE. If the expressions of the SELECT statement do not contain upstream table data, you can execute the SELECT statement without the FROM clause. In this case, the underlying implementation is to select data from an anonymous VALUES table that contains only one row and no columns. This way, you can test user-defined functions (UDFs) or other functions without the need to manually create DUAL tables.

```
-- Create a partitioned table named srcp.
create table if not exists srcp (key string,value bigint) partitioned by (p string);
-- Insert data into the srcp table.
insert into table srcp partition (p) select abs(-1), length('abc'), getdate();
-- Query data from the srcp table.
select * from srcp;
```

The following result is returned:

```
+-----+-----+-----+
| key  | value | p    |
+-----+-----+-----+
| 1    | 3     | 2020-11-25 18:39:48 |
+-----+-----+-----+
```

Support for expressions that do not contain constants

Constant expressions are supported in VALUES expressions. Expressions that do not contain constants are also supported in VALUES expressions. Syntax:

```
select * from values (udf(1)),(to_date('20190101','yyyyMMdd')), (getdate()) t(d);
```

1.6.5.2. SELECT

1.6.5.2.1. SELECT operations

This topic describes how to use SELECT statements for SELECT operations.

Syntax:

```
SELECT [ALL | DISTINCT] select_expr, select_expr, ...
FROM table_reference
[WHERE where_condition]
[GROUP BY col_list]
[ORDER BY order_condition]
[DISTRIBUTE BY distribute_condition [SORT BY sort_condition] ]
[LIMIT number];
```

Limits

- A SELECT statement can return a maximum of 10,000 rows of results. However, no limits are imposed when the SELECT statement is used as a clause. If the SELECT statement is used as a clause, the clause returns all results in response to the query from the upper layer.
- You are not allowed to perform full table scans on partitioned tables by using SELECT statements. To perform a full table scan on a partitioned table, add `set odps.sql.allow.fullscan=true;` before the SQL statement that is used for the full table scan. Then, commit and run the added command with the SQL statement. To enable the full table scan for the entire project, run the following command:

```
setproject odps.sql.allow.fullscan=true;
```

Column expression (select_expr)

Each select_expr expression indicates a column from which you want to read data. You can use the following select_expr expressions to read data from a table.

- Specify the names of the columns from which you want to read data.

```
select shop_name from sale_detail;
```

- Use an asterisk (*) to represent all columns.

```
select * from sale_detail;
-- You can also use a WHERE clause to specify filter conditions.
select * from sale_detail where shop_name like 'hang%';
```

- Use a regular expression.
- Use DISTINCT before the name of a column to filter out duplicate values from the column and return only distinct values. If you use ALL before the name of a column, all values including the duplicate values in the column are returned. If DISTINCT is not used, the default value ALL is used.

```
-- Query data from the region column in the sale_detail table and return only distinct values.
select distinct region from sale_detail;
```

The following result is returned:

```
+-----+
| region |
+-----+
| shanghai |
+-----+
```

```
-- If you use DISTINCT in a SELECT statement where multiple columns are specified, DISTINCT applies to all the columns you specified, instead of a single column.
```

```
select distinct region, sale_date from sale_detail;
```

The following result is returned:

```
+-----+-----+
| region | sale_date |
+-----+-----+
| shanghai | 20191110 |
+-----+-----+
```

Information about a queried destination table (table_reference)

table_reference specifies the table that you want to query. Nested subqueries are also supported in table_reference. Example:

```
select * from (select region from sale_detail) t where region = 'shanghai';
```

WHERE

The following table describes the operators that are used as filter conditions supported by the WHERE clause.

Operator	Description
>, <, =, ≥, ≤, and <>	The relational operators.
LIKE and RLIKE	The Source and Pattern parameters of LIKE and RLIKE must be of the STRING type.
IN and NOT IN	If a subquery is added after the IN or NOT IN operator, the values in only one column can be returned for the subquery and the number of return values cannot exceed 1,000.
BETWEEN...AND	The operator that specifies the query range.

In the WHERE clause of a SELECT statement, you can specify the partitions that you want to scan in a table to avoid a full table scan.

```
SELECT sale_detail.*
FROM sale_detail
WHERE sale_detail.sale_date >= '2008'
AND sale_detail.sale_date <= '2014';
```

User-defined functions (UDFs) support partition pruning. UDFs are executed as small jobs and then replaced with the execution results. You can use one of the following methods to implement UDF-based partition pruning:

- Add an annotation to the UDF class when you write a UDF.

```
@com.aliyun.odps.udf.annotation.UdfProperty(isDeterministic=true)
```

 **Notice** com.aliyun.odps.udf.annotation.UdfProperty defines that the version of referenced odps-sdk-udf must be 0.30.x or later in odps-sdk-udf.jar.

- Add the `set odps.sql.udf.ppr.deterministic = true;` flag before SQL statements. Then, all UDFs in the SQL statements are considered deterministic.

Note This method backfills partitions with execution results. A maximum of 1,000 partitions can be backfilled with execution results. If you add an annotation to a UDF class, an error may be returned, which indicates that more than 1,000 partitions are backfilled with execution results. If you want to ignore the error, add the `set odps.sql.udf.ppr.to.subquery=false;` flag to globally disable this feature. After this feature is disabled, UDF-based partition pruning no longer takes effect.

The WHERE clause in an SQL statement can include the BETWEEN...AND operator. Example:

```
SELECT sale_detail.*
FROM sale_detail
WHERE sale_detail.sale_date BETWEEN '2008' AND '2014';
```

GROUP BY

In most cases, the GROUP BY clause is used with aggregate functions. If a SELECT statement contains aggregate functions, the following rules apply:

- During the parsing of SQL statements, the GROUP BY clause precedes a SELECT operation. Therefore, the value of the GROUP BY clause must be the names of the columns in the input table for the SELECT operation. The value can also be an expression composed of the columns in the input table. The value cannot be the aliases of the output columns from the SELECT operation.
- The value of the GROUP BY clause may be both a column name or an expression composed of input table columns and the alias of an output column from the SELECT operation. In this case, the column name or expression is used as the value of the GROUP BY clause.
- If the `set hive.groupby.position.alias=true;` flag is added before a SELECT statement, the integer constants in GROUP BY are considered column numbers in the SELECT operation.

```
set hive.groupby.position.alias=true;
-- Run this command with the next SELECT statement.
select region, sum(total_price) from sale_detail group by 1;
-- 1 indicates the region column, which is the first column read by the SELECT statement. This statement groups the table data based on the values in the region column and returns distinct region values and total sales of each group.
```

Examples:

```
-- The statement uses the name of the region column in the input table as the value of the GROUP BY clause and groups the table data based on the values in the region column.
select region from sale_detail group by region;
-- The statement groups the table data based on the values in the region column and returns the total sales of each group.
select sum(total_price) from sale_detail group by region;
-- The statement groups the table data based on the values in the region column and returns distinct values and total sales of each group.
select region, sum(total_price) from sale_detail group by region;
-- An error is returned because the alias of the column in the SELECT operation is used as the value of the GROUP BY clause.
select region as r from sale_detail group by r;
-- A complete expression of the column is required.
select 2 + total_price as r from sale_detail group by 2 + total_price;
-- An error is returned because all the columns that do not use aggregate functions in the SELECT operation must be included in the GROUP BY clause.
select region, total_price from sale_detail group by region;
-- The statement can be successfully executed because all the columns that do not use aggregate functions are included in the GROUP BY clause.
select region, total_price from sale_detail group by region, total_price;
```

ORDER BY|DISTRIBUTE BY|SORT BY

- ORDER BY: This clause is used to sort all data records based on the specified columns.
 - To sort data records in descending order, use the desc keyword. By default, data records are sorted in ascending order.
 - When you use ORDER BY to sort data records, NULL is considered the smallest value. This rule is consistent with MySQL, but is different from Oracle.
 - The columns in the ORDER BY clause must be the aliases of the columns in the SELECT operation. If you want to query a column but the column alias is not specified in the SELECT operation, the column name is used as the column alias.
 - If the `set hive.orderby.position.alias=true;` flag is added before SQL statements, integer constants in the ORDER BY clause are considered column numbers in a SELECT operation.

```
-- Set the flag.
set hive.orderby.position.alias=true;
-- Create a table named src.
create table src(key BIGINT,value BIGINT);
-- Query the src table and sort the records that are returned in ascending order by value.
SELECT * FROM src ORDER BY 2 limit 100;
The preceding statement is equivalent to the following statement:
SELECT * FROM src ORDER BY value limit 100;
```

- You can use the OFFSET clause with the ORDER BY LIMIT clause to skip the number of rows specified by OFFSET.

```
-- Sort the rows of the src table in ascending order by key, and return row 11 to row 30. OFFSET 10 indicates that the first 10 rows are skipped, and LIMIT 20 indicates that a maximum of 20 rows can be returned.
SELECT * FROM src ORDER BY key LIMIT 20 OFFSET 10;
```

Examples:

```
-- Query the sale_detail table and sort the first 100 records that are returned in ascending order by region.
SELECT * FROM sale_detail ORDER BY region LIMIT 100;
-- By default, the LIMIT clause is used with the ORDER BY clause. If only the ORDER BY clause is used, an error is returned.
SELECT * FROM sale_detail ORDER BY region;
-- ORDER BY is followed by a column alias.
SELECT region AS r FROM sale_detail ORDER BY region LIMIT 100;
SELECT region AS r FROM sale_detail ORDER BY r LIMIT 100;
```

- **DISTRIBUTE BY:** This clause is used to shard data based on hash values of specific columns. You must specify the alias of an output column from the SELECT operation as the value of the DISTRIBUTE BY clause.

Examples:

```
-- The statement queries values in the region column of the sale_detail table and shards data based on the hash values of the region column.
SELECT region FROM sale_detail DISTRIBUTE BY region;
-- The statement can be successfully executed because the column name is used as the alias.
SELECT region AS r FROM sale_detail DISTRIBUTE BY region;
The preceding statement is equivalent to the following statement:
SELECT region AS r FROM sale_detail DISTRIBUTE BY r;
```

- **SORT BY:** This clause is used to sort specific data records. You can add a keyword, such as ASC or DESC, to sort specific data records. If ASC is used, data records are sorted in ascending order. If DESC is used, data records are sorted in descending order. If SORT BY is not followed by a keyword, data records are sorted in ascending order by default.
 - If the SORT BY clause is preceded by the DISTRIBUTE BY clause, the SORT BY clause sorts specific results of the DISTRIBUTE BY clause.

The DISTRIBUTE BY clause determines how to distribute the output values of the map stages to reducers. To prevent duplicate output values from being distributed to different reducers or to process the same group of data together, you can use the DISTRIBUTE BY clause. This ensures that the same group of data is distributed to the same reducer. Then, use SORT BY to sort the group of data. Example:

```
-- The statement queries values in the region column of the sale_detail table, shards data based on the hash values of the region column, and then sorts the sharded data.
SELECT region FROM sale_detail DISTRIBUTE BY region SORT BY region;
```

- If the SORT BY clause is not preceded by the DISTRIBUTE BY clause, the SORT BY clause sorts the data of each reducer. This process sorts specific data records. This ensures that the output values of each reducer are sorted and increases the data compression ratio. If data is filtered during data reading, the amount of data read from disks is reduced, which makes subsequent global sorting more efficient. Example:

```
SELECT region FROM sale_detail SORT BY region;
```

Note

- The value of ORDER BY, DISTRIBUTE BY, or SORT BY must be the alias of an output column from the SELECT operation. The column alias can be Chinese.
- During the parsing of MaxCompute SQL statements, the ORDER BY, DISTRIBUTE BY, or SORT BY clause is executed after the SELECT operation. Therefore, the value of ORDER BY, DISTRIBUTE BY, or SORT BY must be the alias of an output column from the SELECT operation.
- The ORDER BY clause cannot be used with the DISTRIBUTE BY or SORT BY clause. The GROUP BY clause cannot be used with the DISTRIBUTE BY or SORT BY clause.

LIMIT

The number in the LIMIT clause is a constant that limits the number of rows to return.

- Remove the limit on the simultaneous execution of the ORDER BY and LIMIT clauses

The ORDER BY clause needs to sort all data of a single node. By default, the ORDER BY clause is used with the LIMIT clause to prevent a single node from processing large amounts of data. You can remove the limit on the simultaneous execution of the ORDER BY and LIMIT clauses for a project or session.

- To remove the limit for a project, run the `setproject odps.sql.validate.orderby.limit=false;` command.
- To remove the limit for a session, run the `setproject odps.sql.validate.orderby.limit=false;` command with the SQL statement.

 **Note** If a single node has large amounts of data to sort after the limit is removed, more resources and time are required.

- Limits on the number of rows to return

If you use a SELECT statement without the LIMIT clause or the number in the LIMIT clause exceeds the specified upper limit, you can view only the rows within the upper limit.

The upper limit on the number of rows to return varies based on projects. You can use one of the following methods to configure the upper limit:

- If project protection is disabled, use_instance_tunnel is set to true, and instance_tunnel_max_record is not specified, no limits are imposed on the number of rows to return. For more information about project protection, see [Data protection](#). For more information about the configurations of use_instance_tunnel=true and instance_tunnel_max_record, see [Tunnel operations](#).
- If project protection is enabled, the number of rows to return is limited by READ_TABLE_MAX_ROW. The maximum value of this parameter is 10000.

 **Note** You can run the `show SecurityConfiguration;` command to view the configuration of ProjectProtection. If ProjectProtection is set to true, you can determine whether to disable project protection. You can run the `set ProjectProtection=false;` command to disable project protection. ProjectProtection is disabled by default. For more information about project protection, see [Data protection](#).

1.6.5.2.2. Subquery

This topic describes how to use a SELECT statement to perform subqueries.

A common SELECT statement reads data from multiple tables, such as, `select column_1, column_2 ... from table_name`. The query object can be another SELECT operation, which is a subquery.

Syntax:

```
select * from (select shop_name from sale_detail) a;
```

 **Notice** A subquery must have an alias.

Examples:

```
create table shop as select * from sale_detail;
select a.shop_name, a.customer_id, a.total_price from
(select * from shop) a join sale_detail on a.shop_name = sale_detail.shop_name;
```

 **Note** In a FROM clause, a subquery can be used as a table, which supports a JOIN operation with other tables or subqueries.

1.6.5.2.3. UNION ALL

This topic describes how to use a SELECT statement to perform UNION ALL operations.

Syntax:

```
select_statement union all select_statement
```

 **Note** The UNION ALL clause is used to combine two or more datasets returned from a SELECT operation into one dataset. If duplicate rows exist in the results, all rows that meet the condition are returned, with duplicate rows retained.

MaxCompute SQL does not support UNION ALL between two top-level query statements. To combine the query results of the two statements, you must rewrite the UNION ALL clause as a subquery.

Syntax before rewriting:

```
select * from sale_detail where region = 'hangzhou'
union all
select * from sale_detail where region = 'shanghai';
```

Syntax after rewriting:

```
select * from (
select * from sale_detail where region = 'hangzhou' union all
select * from sale_detail where region = 'shanghai') t;
```

The syntax that uses a pair of parentheses to specify the priority of UNION ALL is supported.

Examples:

```
SELECT * FROM src UNION ALL (SELECT * FROM src2 UNION ALL SELECT * FROM src3);
-- Execute the UNION ALL clause for the src2 and src3 tables. Then, execute the UNION ALL clause for the src table and the result of the first UNION ALL operation.
```

Notice

- For a UNION ALL operation, all subqueries must have the same number of columns, column names, and column types. If the column names are inconsistent, use column aliases.
- In most cases, MaxCompute allows a UNION ALL operation for a maximum of 256 subqueries. If the limit is exceeded, a syntax error is returned.

1.6.5.2.4. JOIN

This topic describes how to use SELECT statements to perform JOIN operations.

MaxCompute supports multiple JOIN operations in an SQL statement. MaxCompute does not support CROSS JOIN. A CROSS JOIN operation joins two tables without the need to specify conditions in the ON clause.

Syntax:

```
join_table:
  table_reference JOIN table_factor [join_condition]
  | table_reference {LEFT OUTER|RIGHT OUTER|FULL OUTER|INNER|NATURAL} JOIN table_reference join_conditio
n
table_reference:
  table_factor
  | join_table
table_factor:
  tbl_name [alias]
  | table_subquery alias
  | ( table_references )
join_condition:
  ON equality_expression ( AND equality_expression )
```

 **Note** equality_expression indicates an equality expression.

Types of JOIN operations:

- LEFT OUTER JOIN: shortened as LEFT JOIN. It returns all rows in the left table, including the rows that do not match any rows in the right table.

```
SELECT a.shop_name AS ashop, b.shop_name AS bshop FROM shop a
LEFT OUTER JOIN sale_detail b ON a.shop_name=b.shop_name;
-- Both the shop and sale_detail tables have the shop_name column. You must use aliases to distinguish the columns in the SELECT operation.
```

 **Note** If the values in some rows of the right table are duplicate, we recommend that you do not consecutively use LEFT JOIN. If you consecutively use LEFT JOIN, data bloat may occur and interrupt your jobs.

- RIGHT OUTER JOIN: shortened as RIGHT JOIN. It returns all rows in the right table, including rows that do not match any rows in the left table.

```
SELECT a.shop_name AS ashop, b.shop_name AS bshop FROM shop a
RIGHT OUTER JOIN sale_detail b ON a.shop_name=b.shop_name;
```

- FULL OUTER JOIN: shortened as FULL JOIN. It returns all rows in both the left and right tables.

```
SELECT a.shop_name AS ashop, b.shop_name AS bshop FROM shop a
FULL OUTER JOIN sale_detail b ON a.shop_name=b.shop_name;
```

- INNER JOIN: The INNER keyword can be omitted. INNER JOIN only returns rows in which a match between the two tables exists.

```
SELECT a.shop_name FROM shop a INNER JOIN sale_detail b ON a.shop_name=b.shop_name;
SELECT a.shop_name FROM shop a JOIN sale_detail b ON a.shop_name=b.shop_name;
```

- NATURAL JOIN: In a NATURAL JOIN operation, the conditions used to join two tables are automatically determined based on the common fields in the two tables. MaxCompute supports OUTER NATURAL JOIN. You can use the USING clause so that the JOIN operation returns common fields only once.

```
SELECT * FROM src NATURAL JOIN src2;
-- Both the src and src2 tables include the key1 and key2 fields. In this case, the preceding statement is equivalent to the following statement:
SELECT src.key1 AS key1, src.key2 AS key2, src.a1, src.a2, src2.b1, src2.b2 FROM src INNER JOIN src2 ON src.key1 = src2.key1 AND src.key2 = src2.key2;
```

- Implicit JOIN operation: A JOIN operation that is performed without the need to specify the JOIN keyword.

```
SELECT * FROM table1, table2 WHERE table1.id = table2.id;
-- The preceding statement is equivalent to the following statement:
SELECT * FROM table1 JOIN table2 ON table1.id = table2.id;
```

Join conditions: You must use equi-joins and combine conditions by using AND. You can use non-equi joins or combine multiple conditions by using OR in a MAPJOIN operation.

```
-- MaxCompute supports multiple JOIN operations in an SQL statement.
SELECT a.* FROM shop a FULL OUTER JOIN sale_detail b ON a.shop_name=b.shop_name
    FULL OUTER JOIN sale_detail c ON a.shop_name=c.shop_name;
-- MaxCompute does not support non-equi joins and returns an error.
SELECT a.* FROM shop a JOIN sale_detail b ON a.shop_name != b.shop_name;
```

You can use parentheses () to specify the priority of JOIN operations. The JOIN operation enclosed in parentheses () has a high priority.

```
SELECT * FROM src JOIN (src2 JOIN src3 on xxx) ON yyy;
-- In the preceding statement, src2 JOIN src3 is first executed. Then, the JOIN operation is performed for the src table and the result of the first JOIN operation.
```

Examples:

The test_table_a and test_table_b tables are available. You want to query the two tables for the rows whose values in the origin column are equal to those in the id column and whose data timestamp is greater than 20180101. If you use LEFT JOIN, all rows in the left table test_table_a are returned.

```
SELECT s.id
    ,s.name
    ,s.origin
    ,d.value
FROM (SELECT * FROM test_table_a WHERE ds > "20180101") s
LEFT JOIN (SELECT * FROM test_table_b WHERE ds > "20180101") d
ON s.origin = d.id;
```

1.6.5.2.5. MAPJOIN hint

This topic describes how to specify MAPJOIN in a SELECT statement to join a large table with one or more small tables.

Background information

You can explicitly specify MAPJOIN in a SELECT statement to join a large table with one or more small tables. MAPJOIN speeds up your query.

A JOIN operation contains three stages: map, shuffle, and reduce. In most cases, tables are joined in the reduce stage.

MAPJOIN joins tables in the map stage instead of the reduce stage. This reduces data transmission time and system resource consumption and optimizes jobs.

In the map stage, MAPJOIN loads all data in the specified tables into the memory of the program that executes the JOIN operation. The tables specified for MAPJOIN must be small tables, and the total memory occupied by the table data cannot exceed 512 MB.

When a large table is joined with one or more small tables, MAPJOIN loads all data in the specified small tables into the memory of the program that executes the JOIN operation. This way, tables are connected in the map stage to accelerate the execution of the JOIN operation.

Usage

To use MAPJOIN, you must specify the hint `/*+ MAPJOIN(table)*/` in a SELECT statement.

For example, `sale_detail` is a large table whose alias is `b`, and `shop` is a small table whose alias is `a`.

- The following statement is used to perform a common JOIN operation:

```
SELECT a.shop_name,  
       b.customer_id,  
       b.total_price  
FROM shop a JOIN sale_detail b  
ON a.shop_name = b.shop_name;
```

- The following statement is used to perform a JOIN operation with MAPJOIN specified:

```
SELECT /*+ MAPJOIN(a)*/  
       a.shop_name,  
       b.customer_id,  
       b.total_price  
FROM shop a JOIN sale_detail b  
ON a.shop_name = b.shop_name;
```

Limits

- When you use MAPJOIN and reference a small table or a subquery, you must reference the alias of the table or subquery.
- MAPJOIN supports small tables in subqueries.
- The left table in a LEFT OUTER JOIN operation must be a large table.
- The right table in a RIGHT OUTER JOIN operation must be a large table.
- Either the left or right table in an INNER JOIN operation can be a large table.
- MAPJOIN cannot be used in a FULL OUTER JOIN operation.
- For MAPJOIN, you can use non-equijoins or combine conditions by using OR. You can leave the ON condition unspecified or use `MAPJOIN ON 1=1`, such as `SELECT /*+ MAPJOIN(a)*/ a.id FROM shop a JOIN table_name b ON 1=1`, to calculate the Cartesian product. However, this calculation method may increase the data amount.
- MaxCompute allows you to specify a maximum of 128 small tables for MAPJOIN. If you specify more than 128 small tables, a syntax error is returned. Separate small tables with commas (,), such as `/*+MAPJOIN(a,b,c)*/`.
- The total memory occupied by small tables cannot exceed 512 MB. MaxCompute compresses your data before storage. As a result, the data amount of small tables sharply increases after they are loaded into the memory. 512 MB indicates the maximum data amount after small tables are loaded into the memory.

Examples

In a MaxCompute SQL statement, you cannot use complex join conditions, such as non-equijoins or the OR logical operator, in the ON condition for a common JOIN operation. However, you can use them for a JOIN operation with MAPJOIN specified.

```
SELECT /* + MAPJOIN(a) */
  a.total_price,
  b.total_price
FROM shop a JOIN sale_detail b
ON a.total_price < b.total_price OR a.total_price + b.total_price < 500;
```

1.6.5.2.6. SELECT TRANSFORM

1.6.5.2.6.1. Overview

This topic describes the syntax of the SELECT TRANSFORM statement in MaxCompute.

The SELECT TRANSFORM statement allows you to start a specific child process and insert data in the required format into the child process by using standard input. Then, the SELECT TRANSFORM statement parses the standard output of the child process to obtain the output data. The SELECT TRANSFORM statement allows you to run scripts in other programming languages without the need to write user-defined functions (UDFs).

Syntax:

```
SELECT TRANSFORM(arg1, arg2 ...)
(ROW FORMAT DELIMITED (FIELDS TERMINATED BY field_delimiter (ESCAPED BY character_escape))
(LINES SEPARATED BY line_separator)
(NULL DEFINED AS null_value))
USING 'unix_command_line'
(RESOURCES 'res_name' (',' res_name')*)
(AS col1, col2 ...)
(ROW FORMAT DELIMITED (FIELDS TERMINATED BY field_delimiter (ESCAPED BY character_escape))
(LINES SEPARATED BY line_separator) (NULL DEFINED AS null_value))
```

Parameters:

- **SELECT TRANSFORM**: the keyword, which can be replaced with the **MAP** or **REDUCE** keyword. These keywords have the same syntax. We recommend that you use **SELECT TRANSFORM** to make the syntax clearer.
- (**arg1, arg2 ...**): the input data. The data format is similar to that of the **SELECT** statement. In the default format, the results of expressions for each parameter are implicitly converted into a value of the **STRING** type. Then, the parameters are combined by `\t` and passed to the specified child process.

 **Note** The format is configurable. For more information, see **ROW FORMAT**.

- **USING**: the commands that are used to start a child process. Take note of the following points:
 - In most MaxCompute SQL statements, the **USING** clause specifies resources. However, in the **SELECT TRANSFORM** statement, the **USING** clause specifies the commands to start a child process. The **USING** clause is used to ensure compatibility with the Hive syntax.
 - The syntax of the **USING** clause is similar to that of a shell script. However, instead of running the shell script, the **USING** clause creates a child process based on the input commands. Therefore, some shell features, such as input and output redirection, pipe, and loop, cannot be used. A shell script can be used as the commands to start a child process if necessary.
- **RESOURCES**: the resources that the specified child process can access. You can use one of the following methods to specify resources:
 - Use the **RESOURCES** clause. Example: `using 'sh foo.sh bar.txt' resources 'foo.sh','bar.txt'`.

- Add the `set odps.sql.session.resources=foo.sh,bar.txt;` flag before SQL statements to specify resources.

Note This is a global configuration. It allows all SELECT TRANSFORM statements to access the specified resources. Separate multiple resource files with commas (,).

- **ROW FORMAT**: the input and output formats. The syntax includes two ROW FORMAT clauses. The first clause specifies the format of the input data, and the second clause specifies the format of the output data. By default, \t is used as a column delimiter, \n is used as a row delimiter, and \N is used to represent NULL values.

Note

- For field_delimiter, character_escape, and line_separator, only one character is accepted. If you specify a string, the first character in the string takes priority over the others.
- MaxCompute supports syntax in the formats that are specified by Hive, such as inputRecordReader, outputRecordReader, and SerDe. You must enable the Hive-compatible mode to use the syntax. To enable the Hive-compatible mode, add `set odps.sql.hive.compatible=true;` before SQL statements.
- If you specify the syntax that is supported by Hive, such as inputRecordReader and outputRecordReader, the performance of SQL queries may deteriorate.

- **AS**: the output columns.

Note

- If you do not specify the data types for output columns, the STRING type is used.
- The output data is obtained after the standard output of the child process is parsed. If the data is not of the STRING type, the system implicitly calls the CAST function to convert the data into the STRING type. Exceptions may occur during the conversion process.
- You must specify data types for all output columns.
- If you omit the AS keyword, the field before the first \t in the standard output data is the key and all the following parts are values. This is equivalent to AS(key, value).

1.6.5.2.6.2. SELECT TRANSFORM examples

Call shell scripts

This topic describes how to use SELECT TRANSFORM to call shell scripts and provides examples.

Assume that you use a shell script to generate 50 rows of data. The values are from 1 to 50. The following code shows the output of the data field:

```
SELECT TRANSFORM(script) USING 'sh' AS (data)
FROM (
  SELECT 'for i in `seq 1 50`; do echo $i; done' AS script
)t
;
```

You can use shell commands as the input for SELECT TRANSFORM.

Note SELECT TRANSFORM allows you to use the scripts of multiple programming languages, such as AWK, Python, Perl, and shell, as the input to implement simple features. You do not need to compile script files or upload resources, which simplifies development.

You can upload a script to implement complex features. For more information, see [Call Python scripts](#).

Call Python scripts

This topic describes how to use SELECT TRANSFORM to call Python scripts and provides examples.

1. Compile a Python script file. In this example, the file name is myplus.py.

```
#!/usr/bin/env python
import sys
line = sys.stdin.readline()
while line:
    token = line.split('\t')
    if (token[0] == '\\N') or (token[1] == '\\N'):
        print '\\N'
    else:
        print str(token[0]) + '\t' + str(token[1])
    line = sys.stdin.readline()
```

2. Add the Python script file as a resource to MaxCompute.

```
add py ./myplus.py -f;
```

3. Execute SELECT TRANSFORM to call the resource.

```
-- Create a test table.
CREATE TABLE testdata(c1 bigint,c2 bigint);
-- Insert test data into the test table.
INSERT INTO TABLE testdata VALUES (1,4),(2,5),(3,6);
-- Execute the following statement:
SELECT
TRANSFORM (testdata.c1, testdata.c2)
USING 'python myplus.py' resources 'myplus.py'
AS (result1,result2)
FROM testdata;
-- The preceding statement is equivalent to the following statement:
set odps.sql.session.resources=myplus.py;
SELECT TRANSFORM (testdata.c1, testdata.c2)
USING 'python myplus.py'
AS (result1,result2)
FROM testdata;
```

Note In MaxCompute, you can use Python scripts as the input for SELECT TRANSFORM. For example, you can run Python commands to call shell scripts.

```
SELECT TRANSFORM('for i in xrange(1, 50): print i;') USING 'python' AS (data);
```

The following result is returned:

```
+-----+-----+
| result1 | result2 |
+-----+-----+
| 1      | 4      |
|      | NULL   |
| 2      | 5      |
|      | NULL   |
| 3      | 6      |
|      | NULL   |
+-----+-----+
```

Call Java scripts

This topic describes how to use SELECT TRANSFORM to call Java scripts and provides examples.

Java scripts are called in a similar way to Python scripts. In this example, you must prepare a Java script file, export it as a JAR package, and then execute the ADD FILE statement to add the JAR package as a resource to MaxCompute. Then, the resource can be called by using SELECT TRANSFORM.

1. Prepare a Java script file and export it as a JAR package. In this example, the name of the JAR package is Sum.jar. Sample code:

```
package com.aliyun.odps.test;
import java.util.Scanner
public class Sum {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        while (sc.hasNext()) {
            String s = sc.nextLine();
            String[] tokens = s.split("\t");
            if (tokens.length < 2) {
                throw new RuntimeException("illegal input");
            }
            if (tokens[0].equals("\n") || tokens[1].equals("\n")) {
                System.out.println("\n");
            }
            System.out.println(Long.parseLong(tokens[0]) + Long.parseLong(tokens[1]));
        }
    }
}
```

2. Add the JAR package as a resource to MaxCompute.

```
add jar ./Sum.jar -f;
```

3. Execute the SELECT TRANSFORM statement to call the resource.

```
-- Create a test table.
CREATE TABLE testdata(c1 bigint,c2 bigint);
-- Insert test data into the test table.
INSERT INTO TABLE testdata VALUES (1,4),(2,5),(3,6);
-- Execute the SELECT TRANSFORM statement.
SELECT TRANSFORM(testdata.c1, testdata.c2) USING 'java -cp Sum.jar com.aliyun.odps.test.Sum' resources 'Sum.jar' FROM testdata;
-- The preceding statement is equivalent to the following statement:
set odps.sql.session.resources=Sum.jar;
SELECT TRANSFORM(testdata.c1, testdata.c2) USING 'java -cp Sum.jar com.aliyun.odps.test.Sum' FROM testdata;
```

The following result is returned:

```
+-----+
| cnt |
+-----+
| 5 |
| 7 |
| 9 |
+-----+
```

You can use the preceding method to run most Java utilities.

Although UDTF frameworks are provided for Java and Python, SELECT TRANSFORM makes it easier to compile code. SELECT TRANSFORM is a simpler process because it is not subject to dependencies or format requirements and can be called offline. The paths for Java and Python offline scripts can be obtained from the JAVA_HOME and PYTHON_HOME environment variables.

Call the scripts of other programming languages

This topic describes how to use SELECT TRANSFORM to call the scripts of other programming languages and provides examples.

SELECT TRANSFORM supports the programming languages described in the preceding topics. It also supports commonly used UNIX commands and script interpreters, such as AWK and Perl interpreters.

Example on how to call an AWK script to display the second column:

```
SELECT TRANSFORM(*) USING "awk '{print $2}'" as (data) from testdata;
```

Example on how to call a Perl script:

```
SELECT TRANSFORM (testdata.c1, testdata.c2) USING "perl -e 'WHILE($input = <STDIN>){print $input;}'" FROM testdata;
```

 **Note** PHP and Ruby are not deployed in MaxCompute compute clusters. Therefore, you cannot call PHP or Ruby scripts in MaxCompute.

Call scripts in series

This topic describes how to use SELECT TRANSFORM to call scripts in series and provides examples.

SELECT TRANSFORM allows you to call scripts in series. For example, you can use DISTRIBUTE BY and SORT BY to pre-process data.

```
SELECT TRANSFORM(key, value) USING 'cmd2' FROM
(
  SELECT TRANSFORM(*) USING 'cmd1' FROM
  (
    SELECT * FROM data DISTRIBUTE BY col2 SORT BY col1
  ) t DISTRIBUTE BY key SORT BY value
) t2;
```

You can also use the MAP or REDUCE keyword to obtain the same results.

```
@a := select * from data distribute by col2 sort by col1;
@b := map * using 'cmd1' distribute by col1 sort by col2 from @a;
reduce * using 'cmd2' from @b;
```

1.6.5.2.6.3. Performance

This topic describes the advantages of SELECT TRANSFORM and user-defined table-valued functions (UDTFs).

The performance of SELECT TRANSFORM and UDTFs varies based on specific scenarios. In most cases, SELECT TRANSFORM performs better. However, UDTFs perform better as the data amount increases. SELECT TRANSFORM is more suitable for ad hoc data analytics because the development of SELECT TRANSFORM is easier.

The following sections describe the advantages of UDTFs and SELECT TRANSFORM.

Advantages of UDTFs

- For UDTFs, the input parameters and output results support multiple data types. For SELECT TRANSFORM, the child process transfers data based on the standard input and output and processes all data as the STRING type. Therefore, SELECT TRANSFORM requires one more step of data type conversions than UDTFs.
- For SELECT TRANSFORM, data transfer depends on the operating system pipe. The pipe has only a 4 KB cache that cannot be specified. If the operating system pipe is empty or fully occupied, SELECT TRANSFORM cannot respond.
- For UDTFs, the constant parameters do not need to be transferred. However, SELECT TRANSFORM does not support this feature.

Advantages of SELECT TRANSFORM

- SELECT TRANSFORM supports two processes: parent and child processes. UDTFs support only a single process. If the usage of computing resources is high and the data throughput is low, SELECT TRANSFORM can take advantage of the multi-core feature of the server.
- SELECT TRANSFORM calls underlying systems to read and write data during data transfer. This allows SELECT TRANSFORM to provide better performance in data transfer than Java programs.
- SELECT TRANSFORM supports the native code of tools such as AWK. This allows SELECT TRANSFORM to deliver more performance benefits than Java programs.

1.6.5.2.7. GROUPING SETS

1.6.5.2.7.1. Overview

This topic describes GROUPING SETS in MaxCompute.

In some cases, you must execute the UNION ALL clause multiple times to aggregate and analyze data from multiple dimensions. For example, you want to aggregate Column A, aggregate Column B, and then aggregate Column A and Column B. The GROUPING SETS clause is a better choice in these cases.

GROUPING SETS is an extension to the GROUP BY clause in a SELECT statement. GROUPING SETS allows you to group results by using various methods without the need to execute multiple SELECT statements. This allows the MaxCompute engine to generate more efficient execution plans with high performance.

 **Note** Most examples in this topic are demonstrated by using MaxCompute Studio. We recommend that you install MaxCompute Studio before you proceed with subsequent operations. For more information, see [MaxCompute Studio](#).

1.6.5.2.7.2. Example

This topic provides an example on how to execute the GROUPING SETS statement.

Example:

1. Prepare data.

```
create table requests LIFECYCLE 20 as
select * from values
(1, 'windows', 'PC', 'Beijing'),
(2, 'windows', 'PC', 'Shijiazhuang'),
(3, 'linux', 'Phone', 'Beijing'),
(4, 'windows', 'PC', 'Beijing'),
(5, 'ios', 'Phone', 'Shijiazhuang'),
(6, 'linux', 'PC', 'Beijing'),
(7, 'windows', 'Phone', 'Shijiazhuang')
as t(id, os, device, city);
```

2. Execute the GROUPING SETS statement.

```
SELECT os,device, city ,COUNT(*)
FROM requests
GROUP BY os, device, city GROUPING SETS((os, device), (city), ());
```

The following result is returned:

```
+----+-----+-----+-----+
| os | device | city | cnt  |
+----+-----+-----+-----+
| NULL | NULL | NULL | 7    |
| NULL | NULL | Beijing | 4    |
| NULL | NULL | Shijiazhuang | 3    |
| ios | Phone | NULL | 1    |
| linux | PC | NULL | 1    |
| linux | Phone | NULL | 1    |
| windows | PC | NULL | 3    |
| windows | Phone | NULL | 1    |
+----+-----+-----+-----+
```

You can also execute multiple SELECT statements to obtain the same result.

```
SELECT NULL, NULL, NULL, COUNT(*)
FROM requests
UNION ALL
SELECT os, device, NULL, COUNT(*)
FROM requests GROUP BY os, device
UNION ALL
SELECT null, null, city, COUNT(*)
FROM requests GROUP BY city;
```

However, the execution of GROUPING SETS is simpler and more efficient.

Note NULL is used as placeholders for the expressions that are not used in GROUPING SETS. This way, you can perform UNION operations on the result sets.

1.6.5.2.7.3. CUBE and ROLLUP

This topic describes the CUBE and ROLLUP functions for GROUPING SETS.

CUBE and ROLLUP are special GROUPING SETS functions. CUBE lists all the possible combinations of specific columns as grouping sets. ROLLUP aggregates data by level to generate grouping sets.

Examples:

```
GROUP BY CUBE(a, b, c)
-- Equivalent to the following statement:
GROUPING SETS((a,b,c),(a,b),(a,c),(b,c),(a),(b),(c),())
GROUP BY ROLLUP(a, b, c)
-- Equivalent to the following statement:
GROUPING SETS((a,b,c),(a,b),(a), ())
GROUP BY CUBE ( (a, b), (c, d) )
-- Equivalent to the following statement:
GROUPING SETS (
  ( a, b, c, d ),
  ( a, b ),
  ( c, d ),
  ( )
)
GROUP BY ROLLUP ( a, (b, c), d )
-- Equivalent to the following statement:
GROUPING SETS (
  ( a, b, c, d ),
  ( a, b, c ),
  ( a ),
  ( )
)
GROUP BY a, CUBE (b, c), GROUPING SETS ((d), (e))
-- Equivalent to the following statement:
GROUP BY GROUPING SETS (
  (a, b, c, d), (a, b, c, e),
  (a, b, d), (a, b, e),
  (a, c, d), (a, c, e),
  (a, d), (a, e)
)
GROUP BY grouping sets((b), (c),rollup(a,b,c))
-- Equivalent to the following statement:
GROUP BY GROUPING SETS (
  (b), (c),
  (a,b,c), (a,b), (a), ()
)
```

1.6.5.2.7.4. GROUPING and GROUPING_ID

This topic describes the GROUPING and GROUPING_ID functions in GROUPING SETS.

NULL is used as placeholders in the results of GROUPING SETS. As a result, the NULL placeholders cannot be distinguished from the NULL values. To address this issue, MaxCompute provides the GROUPING function.

GROUPING allows you to specify the name of a column as a parameter. If specific rows are aggregated based on the column specified in the GROUPING function, 0 is returned, which indicates that NULL is an input value. Otherwise, 1 is returned, which indicates that NULL is a placeholder in GROUPING SETS.

GROUPING_ID allows you to specify the names of one or more columns as parameters. The GROUPING results of these columns are formed into integers by using bit map.

Examples:

```
SELECT a,b,c,COUNT(*),
GROUPING(a) ga, GROUPING(b) gb, GROUPING(c) gc, GROUPING_ID(a,b,c) groupingid
FROM VALUES (1,2,3) as t(a,b,c)
GROUP BY CUBE(a,b,c);
```

The following result is returned:

```

+-----+-----+-----+-----+-----+-----+-----+
| a | b | c | _c3 | ga | gb | gc | groupingid |
+-----+-----+-----+-----+-----+-----+-----+
| NULL | NULL | NULL | 1 | 1 | 1 | 1 | 7 |
| NULL | NULL | 3 | 1 | 1 | 1 | 0 | 6 |
| NULL | 2 | NULL | 1 | 1 | 0 | 1 | 5 |
| NULL | 2 | 3 | 1 | 1 | 0 | 0 | 4 |
| 1 | NULL | NULL | 1 | 0 | 1 | 1 | 3 |
| 1 | NULL | 3 | 1 | 0 | 1 | 0 | 2 |
| 1 | 2 | NULL | 1 | 0 | 0 | 1 | 1 |
| 1 | 2 | 3 | 1 | 0 | 0 | 0 | 0 |
+-----+-----+-----+-----+-----+-----+-----+

```

By default, the columns that are not used in GROUP BY are filled with NULL. You can use the GROUPING function to obtain more useful data.

```

SELECT
  IF(GROUPING(os) == 0, os, 'ALL') as os,
  IF(GROUPING(device) == 0, device, 'ALL') as device,
  IF(GROUPING(city) == 0, city, 'ALL') as city,
  COUNT(*) as count
FROM requests
GROUP BY os, device, city GROUPING SETS((os, device), (city), ());

```

The following figure shows the returned result.

	os	device	city	count
1	ALL	ALL	ALL	7
2	ALL	ALL	Beijing	4
3	ALL	ALL	Shijiazhuang	3
4	ios	Phone	ALL	1
5	linux	PC	ALL	1
6	linux	Phone	ALL	1
7	windows	PC	ALL	3
8	windows	Phone	ALL	1

1.6.5.2.8. UNION, INTERSECT, and EXCEPT

This topic describes the SQL statements that use the UNION, INTERSECT, and EXCEPT set operators, such as UNION, UNION ALL, UNION DISTINCT, INTERSECT, INTERSECT ALL, INTERSECT DISTINCT, EXCEPT, EXCEPT ALL, and EXCEPT DISTINCT.

Syntax:

```

select_statement UNION ALL select_statement;
select_statement UNION [DISTINCT] select_statement;
select_statement INTERSECT ALL select_statement;
select_statement INTERSECT [DISTINCT] select_statement;
select_statement EXCEPT ALL select_statement;
select_statement EXCEPT [DISTINCT] select_statement;
select_statement MINUS ALL select_statement;
select_statement MINUS [DISTINCT] select_statement;

```

Parameters:

- UNION: the union of two datasets. The UNION operation combines two datasets into one dataset.

- **INTERSECT**: the intersection of two datasets. The INTERSECT operation returns the records contained in both datasets.
- **EXCEPT**: the complement of the second dataset in the first dataset. The EXCEPT operation returns the records that are contained in the first dataset, but not in the second dataset.
- **MINUS**: equivalent to EXCEPT.

Examples:

UNION:

- If UNION ALL is used, all records of the two datasets are returned.

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4) t(a, b)
UNION ALL
SELECT * FROM VALUES (1, 2), (1, 4) t(a, b);
```

The following result is returned:

```
+-----+-----+
| a   | b   |
+-----+-----+
| 1   | 2   |
| 1   | 4   |
| 1   | 2   |
| 1   | 2   |
| 3   | 4   |
+-----+-----+
```

- If multiple UNION ALL clauses are used, parentheses can be used to prioritize the clauses.

```
SELECT * FROM src UNION ALL (SELECT * FROM src2 UNION ALL SELECT * FROM src3);
```

- If UNION is used, duplicate records are not included in the returned records. UNION is equivalent to UNION DISTINCT.

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4) t(a, b)
UNION
SELECT * FROM VALUES (1, 2), (1, 4) t(a, b);
-- The preceding statement is equivalent to the following statement:
SELECT DISTINCT * FROM (<Result of UNION ALL>);
```

The following result is returned:

```
+-----+-----+
| a   | b   |
+-----+-----+
| 1   | 2   |
| 1   | 4   |
| 3   | 4   |
+-----+-----+
```

- Assume that UNION is followed by a CLUSTER BY, DISTRIBUTE BY, SORT BY, ORDER BY, or LIMIT clause. If `set odps.sql.type.system.odps2=false;` is used, the clause works on the result of the last select_statement after the UNION operator. If `odps.sql.type.system.odps2=true;` is used, the clause works on the result of all the UNION operations.

```
set odps.sql.type.system.odps2=true;
SELECT explode(array(3, 1)) AS (a) UNION ALL SELECT explode(array(0, 4, 2)) AS (a) ORDER BY a LIMIT 3;
```

The following result is returned:

```
+-----+
| a |
+-----+
| 0 |
| 1 |
| 2 |
+-----+
```

INTERSECT :

- INTERSECT ALL:

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (5, 6) t(a, b)
INTERSECT ALL
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (5, 7) t(a, b);
```

The following result is returned:

```
+-----+-----+
| a | b |
+-----+-----+
| 1 | 2 |
| 1 | 2 |
| 3 | 4 |
+-----+-----+
```

- INTERSECT DISTINCT :

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (5, 6) t(a, b)
INTERSECT DISTINCT
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (5, 7) t(a, b);
```

The following result is returned. The preceding statement is equivalent to the `SELECT DISTINCT * FROM (< Result of INTERSECT ALL >);` statement.

```
+-----+-----+
| a | b |
+-----+-----+
| 1 | 2 |
| 3 | 4 |
+-----+-----+
```

EXCEPT :

- EXCEPT ALL:

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (3, 4), (5, 6), (7, 8) t(a, b)
EXCEPT ALL
SELECT * FROM VALUES (3, 4), (5, 6), (5, 6), (9, 10) t(a, b);
```

The following result is returned:

a	b
1	2
1	2
3	4
7	8

• EXCEPT DISTINCT:

```
SELECT * FROM VALUES (1, 2), (1, 2), (3, 4), (3, 4), (5, 6), (7, 8) t(a, b)
EXCEPT
SELECT * FROM VALUES (3, 4), (5, 6), (5, 6), (9, 10) t(a, b);
```

The following result is returned. The preceding statement is equivalent to the `SELECT DISTINCT * FROM left_branch EXCEPT ALL SELECT DISTINCT * FROM right_branch;` statement.

a	b
1	2
7	8

 Note

- Set operations do not necessarily return sorted results.
- The two branches for a set operation must have the same number of output columns with consistent data types. If data types are inconsistent, implicit conversion of the data types may be performed. To ensure compatibility for set operations, MaxCompute forbids the implicit conversion between STRING and other data types.
- MaxCompute supports a maximum of 256 branches for a set operation. If the limit is exceeded, an error is returned.

1.6.5.3. EXPLAIN

This topic describes the EXPLAIN statement in MaxCompute SQL.

This statement shows the structure of the execution plan of a DML statement. An execution plan is a program that executes SQL statements.

Syntax:

```
EXPLAIN <DML query>;
```

 Note

The execution results of an EXPLAIN statement include:

- Dependencies among all the jobs of this DML statement
- Dependencies among all the tasks of each job
- Dependencies among all operators in a task

Examples:

```
EXPLAIN
SELECT abs(a.key), b.value FROM src a JOIN src1 b ON a.value = b.value;
```

The output of the EXPLAIN statement consists of the following three parts:

- Job dependencies: `job0 is root job`. This query requires only one job (job0). Therefore, only one row of information is displayed.
- Task dependencies. Example:

```
In Job job0:
root Tasks: M1_Stg1, M2_Stg1
J3_1_2_Stg1 depends on: M1_Stg1, M2_Stg1
```

 Note

- Job0 contains three tasks, M1_Stg1, M2_Stg1, and J3_1_2_Stg1. J3_1_2_Stg1 is executed after the M1_Stg1 and M2_Stg1 tasks are complete.
- Naming conventions for a task: MaxCompute provides the task types of map, reduce, join, and local work. The first letter in a task name indicates the type of the task. For example, M2Stg1 is a map task. The number following the first letter indicates the task ID. This ID is unique among all tasks that correspond to the current query. Digits that are combined by underscores (_) indicate the task dependencies. For example, J3_1_2_Stg1 indicates that the current task with the ID of 3 depends on two tasks whose IDs are 1 and 2.

- Operator structure, where each operator string describes the execution semantics of a task.

```
In Task M1_Stg1:
Data source: yudi_2.src          # Data source describes the input of the task.
TS: alias: a                    # TableScanOperator
RS: order: +                    # ReduceSinkOperator
  keys:
    a.value
  values:
    a.key
  partitions:
    a.value
In Task J3_1_2_Stg1:
JOIN: a INNER JOIN b           # JoinOperator
SEL: Abs(UDFToDouble(a._col0)), b._col5 # SelectOperator
FS: output: None               # FileSinkOperator
In Task M2_Stg1:
Data source: yudi_2.src1
TS: alias: b
RS: order: +
  keys:
    b.value
  values:
    b.value
  partitions:
    b.value
```

The following table describes the operators.

Operator	Description
TableScanOperator	Describes the logic of FROM statement blocks in a query statement. The alias of the input table is displayed in the EXPLAIN results.
SelectOperator	<p>Describes the logic of SELECT statement blocks in a query statement. The columns that are transferred to the next operator are displayed in the EXPLAIN results. Separate multiple columns with commas (,).</p> <ul style="list-style-type: none"> ◦ If columns are referenced, the value is in the < alias >.< column_name > format. ◦ If the result of an expression is transferred, the value is displayed as a function, such as func1(arg1_1, arg1_2, func2(arg2_1, arg2_2)). ◦ If constants are transferred, the value is immediately displayed.
FilterOperator	Describes the logic of WHERE statement blocks in a query statement. The EXPLAIN results include a WHERE clause, with the display rules that are similar to those of SelectOperator.
JoinOperator	Describes the logic of JOIN statement blocks in a query statement. The EXPLAIN results show which tables are joined in which way.
GroupByOperator	Describes the logic of aggregate operations. This operator is displayed if an aggregate function is used in a query. The content of the aggregate function is displayed in the EXPLAIN results.
ReduceSinkOperator	Describes the logic of data distribution between tasks. If the result of the current task is transferred to another task, ReduceSinkOperator must be used to distribute data at the last stage of the current task. The sorting method of output data records, the distributed keys and values, and the columns used to calculate the hash value are displayed in the EXPLAIN results.
FileSinkOperator	Describes the logic of storage operations on final data records. If an INSERT statement block exists in a query, the name of the required table is displayed in the EXPLAIN results.
LimitOperator	Describes the logic of LIMIT statement blocks in a query statement. The number of records specified by LIMIT is displayed in the EXPLAIN results.
MapjoinOperator	Describes JOIN operations on large tables. This operator is similar to JoinOperator.

 **Note**

- If a query statement is complex, the EXPLAIN statement returns excessive results. As a result, API limits are reached and complete results cannot be obtained. In this case, you can split a query and execute the EXPLAIN statement on each subquery to understand the structure of the job.
- If you specify more than 10,000 partitions in a query, large amounts of source data needs to be read. To circumvent this limit, you can add filter conditions in the query to filter out most partitions.

1.6.5.4. IF statement

This topic describes the IF statement that MaxCompute SQL supports.

MaxCompute SQL supports the IF-ELSE statement. You can use the IF-ELSE statement to execute SQL scripts with specific conditions. A condition in the IF-ELSE statement can be a standard variable or a scalar subquery that returns only one column value from one row.

The IF statement allows the system to automatically select the execution logic based on the specified conditions. MaxCompute supports the following IF syntax:

```
IF (condition) BEGIN
  statement 1
  statement 2
  ...
END
IF (condition) BEGIN
  statements
END ELSE IF (condition2) BEGIN
  statements
END ELSE BEGIN
  statements
END
```

 **Note** The BEGIN and END conditional clause can be omitted if it contains only one statement, similar to '{ }' in Java.

The IF statement can contain two types of conditions: expressions and scalar subqueries. Both of them are of the BOOLEAN type.

- Expressions: A BOOLEAN-type expression in the IF-ELSE statement determines which branch is executed at the compiling stage. Example:

```
@date := '20190101';
@row TABLE(id STRING); -- Declare the row variable. The type of the row is Table and schema is STRING.
IF ( cast(@date as bigint) % 2 == 0 ) BEGIN
@row := SELECT id from src1;
END ELSE BEGIN
@row := SELECT id from src2;
END
INSERT OVERWRITE TABLE dest SELECT * FROM @row;
```

- Scalar subqueries: A BOOLEAN-type scalar subquery in the IF-ELSE statement determines which branch is executed at the running stage. Therefore, you must submit multiple jobs. Example:

```
@i bigint;
@t table(id bigint, value bigint);
IF ((SELECT count(*) FROM src WHERE a = '5') > 1) BEGIN
@i := 1;
@t := select @i, @i*2;
END ELSE
BEGIN
@i := 2;
@t := select @i, @i*2;
END
select id, value from @t;
```

1.6.5.5. UPDATE and DELETE

This topic describes the UPDATE and DELETE statements in MaxCompute SQL.

The UPDATE and DELETE statements are used to manage data of specific rows in tables or partitions.

Background information

MaxCompute DML has limits on table operations. It allows you to use only the INSERT OVERWRITE and INSERT INTO statements to delete and update some rows in a table. For example, if you use an INSERT statement to modify small amounts of data in a table or partition, you must use a SELECT statement to read all data and modify the data. Then, you must execute the INSERT statement to insert the modified data into the table or partition. This is a low-efficiency process. However, if you use the UPDATE and DELETE statements, the amount of data read or write in a query significantly decreases.

History tables were considered the best to perform multiple UPDATE operations at a time. If you use a history table, you must add auxiliary columns such as start_date and end_date in the table. These columns indicate the lifecycle of the data records in a row. To query the current status of a table, you must find the latest status from large amounts of data based on the timestamp. This is a time-consuming process. However, the UPDATE and DELETE statements directly read data from the current table.

Description

When you execute a DELETE statement, the system automatically generates a Delta file. txnid(bigint) and rowid(bigint) in the Delta file indicate which rows in which transactional table are deleted from the Base file.

 **Note** The Delta file stores the row numbers that are used to indicate the deleted rows in the Base file.

If you execute a DELETE statement again, another Delta file named f2.del is generated. The f2.del file also stores the row numbers based on the Base file. During data reading, all Delta files are queried and data that is not deleted is returned.

Similarly, the UPDATE statement is converted into the combination of the DELETE and INSERT INTO statements.

If jobs are concurrently run and the tables on which you want to perform operations are the same, conflicts may occur. For more information, see [ACID semantics of MaxCompute concurrent write jobs](#).

Usage notes

- The UPDATE and DELETE operations must be performed based on transactional tables. For more information about how to create a transactional table, see [Create a table](#).
- If you perform UPDATE operations, you must add the `set odps.service.mode=off;` attribute to the UPDATE statements for execution.
- If the ratio of the rows that you want to delete or update to all rows is small, the performance of the data reads from a table is slightly affected. In this case, we recommend that you use the UPDATE or DELETE statement. If the ratio is less than 5%, we recommend that you use the UPDATE or DELETE statement. The maximum ratio varies based on your business scenarios.
- If you perform the UPDATE or DELETE operation multiple times to update or delete a small number of rows in a table, we recommend that you execute the COMPACT statement after the UPDATE and DELETE statements. This reduces the storage usage of the table.
- If you perform less UPDATE or DELETE operations to update or delete a large number of rows in a table and will frequently read data from the table, we recommend that you use the INSERT OVERWRITE and INSERT INTO statements. For example, a total of 10% of the data in a table needs to be updated or deleted ten times a day. In this case, we recommend that you evaluate the sum of the costs of the UPDATE or DELETE

statement and the read performance consumption based on the generated Delta file. Then, compare it with that of the INSERT OVERWRITE and INSERT INTO statements to select an efficient method.

- After you delete data, a Delta file is generated. As a result, the storage may not be released. If you want to use the DELETE statement to delete data and release the storage, you can use the ALTER TABLE COMPACT statement to merge the Base files with Delta files.

Limits

- If you use the ALTER TABLE COMPACT statement in MaxCompute SQL mode of DataWorks, a compatibility issue occurs. To handle this issue, we recommend that you use the MaxCompute client whose version must be **0.35.0**.
- You can specify the transactional attribute for a table only when you create the table. However, you are not allowed to modify the transactional attribute of an existing table by using the ALTER TABLE COMPACT statement. If you modify the transactional attribute of an existing table, an error is returned.

```
alter table not_txn_tbl set tblproperties("transactional"="true");
-- The error message is FAILED: Catalog Service Failed, Error Code: 151, Error Message: Set transactional is not supported.
```

- You cannot specify clustered tables or external tables as transactional tables.
- Existing internal tables or external tables cannot be converted into transactional tables, and transactional tables cannot be converted into standard tables.
- Tasks from peripheral systems, such as Graph and Machine Learning Platform for AI (PAI), are not supported. If these tasks access a transactional table, an error is returned.
- CLONE TABLE, MERGE PARTITION, and incremental replication operations cannot be performed on transactional tables.
- You cannot use Kunneling to back up data in transactional tables. If you want to perform the UPDATE, DELETE, or INSERT OVERWRITE operation on important data in transactional tables, you must back up the data in advance.
- You must have the SELECT and UPDATE permissions on a table to perform the UPDATE and DELETE operations.

DELETE

Syntax:

```
delete from <table_name> [WHERE where_condition];
```

Description: This statement is used to delete some rows from a table.

Examples:

Example 1: Delete some rows from a table.

```
-- Create a transactional table.
create table if not exists acid_delete(id bigint) tblproperties ("transactional"="true");
-- Insert data into the table.
insert overwrite table acid_delete values(1),(2),(3);
-- Check the inserted data.
select * from acid_delete;
```

The following result is returned:

```
+-----+
| id  |
+-----+
| 1   |
| 2   |
| 3   |
+-----+
```

```
-- Delete the row whose id value is 2. You must enter Yes or No to confirm the operation in the MaxCompute client.
delete from acid_delete where id = 2;
-- Read data from the table. Only the rows whose id values are 1 and 3 are retained in the table.
select * from acid_delete;
```

The following result is returned:

```
+-----+
| id  |
+-----+
| 1   |
| 3   |
+-----+
```

Example 2: Delete some rows from a partitioned table.

```
-- Create a transactional table.
create table if not exists acid_delete_pt(id bigint) partitioned by(ds string) tblproperties ("transactional"="true");
-- Add partitions to the table.
alter table acid_delete_pt add if not exists partition (ds='2019');
-- Insert data into the partitions.
insert overwrite table acid_delete_pt partition (ds='2019') values(1),(2),(3);
-- Check the inserted data.
select * from acid_delete_pt where ds = '2019';
```

The following result is returned:

```
+-----+-----+
| id | ds  |
+-----+-----+
| 1  | 2019 |
| 2  | 2019 |
| 3  | 2019 |
+-----+-----+
```

```
-- Delete data from the partitions. You must enter Yes or No to confirm the operation in the MaxCompute client.
delete from acid_delete_pt where ds='2019' and id=2;
-- Read data from the table. Only the rows whose id values are 1 and 3 are retained in the table.
select * from acid_delete_pt where ds = '2019';
```

The following result is returned:

```
+-----+-----+
| id | ds |
+-----+-----+
| 1 | 2019 |
| 3 | 2019 |
+-----+-----+
```

UPDATE

Syntax:

```
update <table_name> set col1 = value1 [, col2 = value2 ...] [WHERE where_condition];
update <table_name> set (col1 [, col2 ...]) = (value1 [, value2 ...]);
```

Description: This statement is used to update some rows in a table.

 **Note** The UPDATE statement is converted into the combination of the DELETE and INSERT INTO statements to update some rows.

Examples:

Example 1: Update some rows in a table.

```
-- Create a transactional table.
create table if not exists acid_update(id bigint) tblproperties ("transactional"="true");
-- Insert data into the table.
insert overwrite table acid_update values(1),(2),(3);
-- Check the inserted data.
select * from acid_update;
```

The following result is returned:

```
+-----+
| id |
+-----+
| 1 |
| 2 |
| 3 |
+-----+
```

```
-- Update the value 2 in the id column to 4.
set odps.service.mode=off;
update acid_update set id = 4 where id = 2;
-- Check the update result. The value 2 is updated to 4.
select * from acid_update;
```

The following result is returned:

```
+-----+
| id  |
+-----+
| 4   |
| 1   |
| 3   |
+-----+
```

Example 2: Update some rows in a partitioned table.

```
-- Create a transactional table.
create table if not exists acid_update_pt(id bigint) partitioned by(ds string) tblproperties ("transactional"="true");
-- Add partitions to the table.
alter table acid_update_pt add if not exists partition (ds= '2019');
-- Insert data into the partitions.
insert overwrite table acid_update_pt partition (ds='2019') values(1),(2),(3);
-- Check the inserted data.
select * from acid_update_pt where ds = '2019';
```

The following result is returned:

```
+-----+-----+
| id | ds |
+-----+-----+
| 1  | 2019 |
| 2  | 2019 |
| 3  | 2019 |
+-----+-----+
```

```
-- Update the id values of rows in partition 2019 to 4. The rows must have the id values of 2.
set odps.service.mode=off;
update acid_update_pt set id = 4 where ds = '2019' and id = 2;
-- Check the update result. The value 2 is updated to 4.
select * from acid_update_pt where ds = '2019';
```

The following result is returned:

```
+-----+-----+
| id | ds |
+-----+-----+
| 4  | 2019 |
| 1  | 2019 |
| 3  | 2019 |
+-----+-----+
```

Example 3: Update several columns and update a destination table by using the data in a source table.

```

-- Create a destination table named acid_update_t and create a table named acid_update_s for joins.
create table if not exists acid_update_t(id bigint,value1 bigint,value2 bigint) tblproperties ("transactional"="true");
create table if not exists acid_update_s(id bigint,value1 bigint,value2 bigint);
-- Insert data into these tables.
insert overwrite table acid_update_t values(2,20,21),(3,30,31),(4,40,41);
insert overwrite table acid_update_s values(1,100,101),(2,200,201),(3,300,301);
--1. Use constants to update the acid_update_t table.
set odps.service.mode=off;
update acid_update_t set (value1, value2) = (3, 4);
--2. Use the data in the acid_update_s table to update the acid_update_t table. The acid_update_t table is left joined with the acid_update_s table. If you want to update only the intersection of the two tables, you must add the WHERE clause in update <table_name>.
set odps.service.mode=off;
update acid_update_t set (value1, value2) = (select value1, value2 from acid_update_s where acid_update_t.id = acid_update_s.id);
--3. Use the data in the acid_update_s table to update the acid_update_t table. Add filter conditions for the acid_update_t table to update only the intersection of the two tables.
set odps.service.mode=off;
update acid_update_t set (value1, value2) = (select value1, value2 from acid_update_s where acid_update_t.id = acid_update_s.id) where acid_update_t.id in (select id from acid_update_s);
--4. Use the aggregate results of the acid_update_t and acid_update_s tables to update the acid_update_t table.
set odps.service.mode=off;
update acid_update_t set (id, value1, value2) = (select id, max(value1),max(value2) from acid_update_s where acid_update_t.id = acid_update_s.id group by acid_update_s.id) where acid_update_t.id in (select id from acid_update_s);

```

Merge Into

The MERGE INTO statement encapsulates the INSERT, UPDATE, and DELETE operations. You can use this statement to perform multiple operations on a destination table after the table is joined with a source table.

MaxCompute DML provides the UPDATE and DELETE statements to enrich its expressions. However, if you want to use several INSERT, UPDATE, and DELETE statements to perform multiple UPDATE operations on a destination table at a time, you must execute several statements, which triggers multiple full table scans. The MERGE INTO statement requires only one full table scan for you to complete the operations. The MERGE INTO statement is more efficient than the INSERT, UPDATE, and DELETE statements.

If you use the INSERT, UPDATE, and DELETE statements in a job to update a table, the update succeeds only when all the statements are successfully executed. If only some of these statements are successfully executed, the execution cannot be rolled back. However, the MERGE INTO statement is atomic, which can be used to prevent this issue.

Syntax:

```

MERGE INTO <target table> AS T USING <source expression/table> AS S
ON <boolean expression1>
WHEN MATCHED [AND <boolean expression2>] THEN UPDATE SET <set clause list>
WHEN MATCHED [AND <boolean expression3>] THEN DELETE
WHEN NOT MATCHED [AND <boolean expression4>] THEN INSERT VALUES <value list>

```

Description: This statement is used to update or insert the data from source expression/table into target table.

Parameters:

- source expression/table: the source table. The value of this parameter can be a table, view, and subquery.
- target table: the destination table, which must be an existing physical table.

- ON: the clause that is used to determine whether the source table is joined with the destination table. boolean expression specifies the join condition.
- WHEN MATCHED...THEN: specifies action when the ON clause determines that the two tables are joined. Data obtained from multiple WHEN MATCHED [AND <boolean expression>] clauses must have no intersections. The MERGE INTO statement has clear objectives. You are not allowed to perform the INSERT or UPDATE operation on the same row in a single MERGE INTO statement. When you execute the MERGE INTO statement, make sure that the data in the source tables that meet the ON joint condition is unique. Otherwise, an error is returned.

 Note

- If a MERGE INTO statement has three WHEN clauses, the UPDATE, DELETE, and INSERT operations can each appear at most once in each clause.
- WHEN NOT MATCHED must be the last WHEN clause and can be used only for an INSERT operation.
- If both the UPDATE and DELETE statements are included in a MERGE INTO statement, the statement that appears first must include [AND <boolean expression>].

Examples:

Prepare the test data.

```
-- Create a destination table.
create table if not exists acid_address_book_base1
(id bigint,first_name string,last_name string,phone string)
partitioned by(year string, month string, day string, hour string)
tblproperties ("transactional"="true");
-- Create a source table.
create table if not exists tmp_table1
(id bigint, first_name string, last_name string, phone string, _event_type_string);
-- Insert the test data into the destination table.
insert overwrite table acid_address_book_base1
partition(year='2020', month='08', day='20', hour='16')
values (4, 'nihaho', 'li', '222'), (5, 'tahao', 'ha', '333'),
(7, 'djh', 'hahh', '555');
-- Insert the test data into the source table.
insert overwrite table tmp_table1 values
(1, 'hh', 'liu', '999', 'I'), (2, 'cc', 'zhang', '888', 'I'),
(3, 'cy', 'zhang', '666', 'I'),(4, 'hh', 'liu', '999', 'U'),
(5, 'cc', 'zhang', '888', 'U'),(6, 'cy', 'zhang', '666', 'U');
```

If data meets the ON joint condition and is from the source table, you can use the data to perform the MERGE INTO operation on the destination table. If data does not meet the ON joint condition but is of the I type specified by event_type in the source table, you can insert the data into the destination table.

```
merge into acid_address_book_base1 as t using tmp_table1 as s
on s.id = t.id and t.year='2020' and t.month='08' and t.day='20' and t.hour='16'
when matched
then update set t.first_name = s.first_name, t.last_name = s.last_name, t.phone = s.phone
when not matched and (s._event_type_='I')
then insert values(s.id, s.first_name, s.last_name, s.phone, '2020', '08', '20', '16');
-- Query the destination table after the preceding operations are complete.
select * from acid_address_book_base1;
```

The following result is returned:

```

+-----+-----+-----+-----+-----+-----+
| id   | first_name | last_name | phone | year | month | day | hour |
+-----+-----+-----+-----+-----+-----+-----+
| 4   | hh   | liu   | 999 | 2020 | 08   | 20 | 16 |
| 5   | cc   | zhang | 888 | 2020 | 08   | 20 | 16 |
| 7   | djh  | hahh  | 555 | 2020 | 08   | 20 | 16 |
| 1   | hh   | liu   | 999 | 2020 | 08   | 20 | 16 |
| 2   | cc   | zhang | 888 | 2020 | 08   | 20 | 16 |
| 3   | cy   | zhang | 666 | 2020 | 08   | 20 | 16 |
+-----+-----+-----+-----+-----+-----+

```

ALTER TABLE COMPACT

If you perform DELETE operations on a transactional table, the Base file that you want to delete is not modified. A Delta file is generated each time you perform a DELETE operation. As a result, if you perform multiple DELETE operations, the generated Delta files occupy more storage usage.

If you perform several DELETE operations on the same table or partition, a large number of Delta files may be generated based on the amount of the data that you delete. Similarly, if you perform an UPDATE operation which is converted into the combination of the DELETE and INSERT INTO operations, Delta files are also generated. If the system reads data from the table, it must load the generated Delta files to find the deleted rows. A large number of Delta files reduce data reading efficiency.

In this case, you can use the ALTER TABLE COMPACT statement to merge the Base files with Delta files to reduce storage usage and improve data reading efficiency.

Syntax:

```
alter table <table_name> [partition (partition_key = 'partition_value' [, ...])] compact [major|minor];
```

Description: This statement is used to merge the Base files with Delta files to reduce storage usage and improve data reading efficiency.

- Minor Compaction: merges all Delta files with their Base files. The Delta files must be generated based on the same Base file. Delta files that are generated based on different Base files cannot be merged.
- Major Compaction: merges all Delta files generated based on the same Base file with the Base file, deletes the Delta files, and merges small files in multiple Base files that correspond to tables. If the size of the Base file is less than 32 MB or a Delta file is generated, the INSERT OVERWRITE operation is performed on the table. However, if the size of the Base file is greater than or equal to 32 MB or no Delta files are generated, the INSERT OVERWRITE operation is not performed on the table.

Examples:

```
alter table acid_delete compact minor;
alter table acid_update_pt partition (ds = '2019') compact major;
```

The following result is returned:

```
Summary:
table name: acid_update_pt/ds=2019 instance count: 2 run time: 16
before merge, file count: 8 file size: 2423 file physical size: 7269
after merge, file count: 2 file size: 642 file physical size: 1926
```

1.6.6. Built-in functions

1.6.6.1. Mathematical functions

1.6.6.1.1. ABS

This topic describes the ABS function in mathematical functions and provides examples.

Syntax:

```
double abs(double number)
bigint abs(bigint number)
decimal abs(decimal number)
```

Description: This function calculates the absolute value of number.

Parameters: The number parameter supports a value of the BIGINT, DOUBLE, or DECIMAL type. If the input value is of the BIGINT, DOUBLE, or DECIMAL type, a value of the same type is returned. If the input value is of the STRING type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: The type of the return value depends on that of the input value, which can be DOUBLE, BIGINT, or DECIMAL. If the input value is NULL, NULL is returned.

 **Note** If the input value is of the BIGINT type and is greater than the maximum value of the BIGINT type, a value of the DOUBLE type is returned. However, the precision may be lost.

Examples:

```
abs(null)=null
abs(-1)=1
abs(-1.2)=1.2
abs("-2")=2.0
abs(122320837456298376592387456923748)=1.2232083745629837e32
```

The following example shows the usage of an ABS function in SQL statements. Other built-in functions, except window functions and aggregate functions, are used in a similar way.

```
select abs(id) from tbl1;
-- Calculate the absolute value of the id field in tbl1.
```

1.6.6.1.2. ACOS

Function declaration:

```
double acos(double number)
decimal acos(decimal number)
```

Purpose: It is used to calculate the arccosine of a number.

Description:

number: double or decimal type. Value range: -1 to 1. If the input is of the string or bigint type, it is implicitly converted into a value of the double type before this computation. For all other input types, an error is returned.

Returned value: double or decimal type. Value range: 0 to π . If number is NULL, NULL is returned.

Example:

```
acos("0.87") = 0.5155940062460905  
acos(0) = 1.5707963267948966
```

1.6.6.1.3. ASIN

This topic describes the ASIN function in mathematical functions and provides examples.

Syntax:

```
double asin(double number)  
decimal asin(decimal number)
```

Description: This function calculates the arcsine of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. The value ranges from -1 to 1. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. The value ranges from $-\pi/2$ to $\pi/2$. If the input value is NULL, NULL is returned.

Examples:

```
asin(1) = 1.5707963267948966  
asin(-1) = -1.5707963267948966
```

1.6.6.1.4. ATAN

This topic describes the ATAN function in mathematical functions and provides examples.

Syntax:

```
double atan(double number)
```

Description: This function calculates the arctangent of number.

Parameters: The number parameter supports a value of the DOUBLE type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE type is returned. The value ranges from $-\pi/2$ to $\pi/2$. If the input value is NULL, NULL is returned.

Examples:

```
atan(1) = 0.7853981633974483;  
atan(-1) = -0.7853981633974483
```

1.6.6.1.5. CEIL

This topic describes the CEIL function in mathematical functions and provides examples.

Syntax:

```
bigint ceil(double value)  
bigint ceil(decimal value)
```

Description: This function rounds up value and returns the nearest integer.

Parameters: The value parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If the input value is NULL, NULL is returned.

Examples:

```
ceil(1.1) = 2  
ceil(-1.1) = -1
```

1.6.6.1.6. CONV

This topic describes the CONV function in mathematical functions and provides examples.

Syntax:

```
string conv(string input, bigint from_base, bigint to_base)
```

Description: This function converts a number from one number system to another.

Parameters:

- input: the integer you want to convert, which is of the STRING type. If the input value is of the BIGINT or DOUBLE type, it is implicitly converted to a value of the STRING type before calculation.
- from_base, to_base: decimal numbers. The values can be 2, 8, 10, or 16. If the input value is of the STRING or DOUBLE type, it is implicitly converted to a value of the BIGINT type before calculation.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned. The conversion process runs at 64-bit precision. If an overflow occurs, an error is returned. If the input value is a negative value that begins with an en dash (-), an error is returned. If the input value is a decimal, it is converted to an integer before the conversion of number systems. The decimal part is left out.

Examples:

```
conv('1100', 2, 10) = '12'  
conv('1100', 2, 16) = 'c'  
conv('ab', 16, 10) = '171'  
conv('ab', 16, 16) = 'ab'
```

1.6.6.1.7. COS

This topic describes the COS function in mathematical functions and provides examples.

Syntax:

```
double cos(double number)  
decimal cos(decimal number)
```

Description: This function calculates the cosine of number, which is a radian value.

Parameters: number, a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

Examples:

```
cos(3.1415926/2) = 2.6794896585028633e-8  
cos(3.1415926) = -0.99999999999999986
```

1.6.6.1.8. COSH

This topic describes the COSH function in mathematical functions.

Syntax:

```
double cosh(double number)  
decimal cosh(decimal number)
```

Description: This function calculates the hyperbolic cosine of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.9. COT

This topic describes the COT function in mathematical functions.

Syntax:

```
double cot(double number)  
decimal cot(decimal number)
```

Description: This function calculates the cotangent of number, which is a radian value.

Parameters: number, a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.10. EXP

This topic describes the EXP function in mathematical functions.

Syntax:

```
double exp(double number)  
decimal exp(decimal number)
```

Description: This function calculates the exponential value of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: The exponential value of number is returned. The value is of the DOUBLE or DECIMAL type. If the input value is NULL, NULL is returned.

1.6.6.1.11. FLOOR

This topic describes the FLOOR function in mathematical functions and provides examples.

Syntax:

```
bigint floor(double number)
bigint floor(decimal number)
```

Description: This function rounds down number and returns the nearest integer that is no greater than the value of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If the input value is NULL, NULL is returned.

Examples:

```
floor(1.2) = 1
floor(1.9) = 1
floor(0.1) = 0
floor(-1.2) = -2
floor(-0.1) = -1
floor(0.0) = 0
floor(-0.0) = 0
```

1.6.6.1.12. LN

This topic describes the LN function in mathematical functions.

Syntax:

```
double ln(double number)
decimal ln(decimal number)
```

Description: This function calculates the natural logarithm of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned. If the input value is a negative value or 0, an error is returned.

Examples:

```
select ln(1000);
```

The following result is returned:

```
+-----+
|_c0  |
+-----+
|6.907755278982137|
+-----+
```

1.6.6.1.13. LOG

This topic describes the LOG function in mathematical functions.

Syntax:

```
double log(double base, double x)
decimal log(decimal base, decimal x)
```

Description: This function calculates the logarithm of x whose base number is base.

Parameters:

- **base**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.
- **x**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: The logarithm value of the DOUBLE or DECIMAL type is returned. If an input value is NULL, NULL is returned. If an input value is a negative value or 0, an error is returned. If the value of base is 1, an error is returned. The value 1 causes division by zero.

Examples:

```
select log(10,100);
```

The following result is returned:

```
+-----+
|_c0  |
+-----+
|2.0  |
+-----+
```

1.6.6.1.14. POW

This topic describes the POW function in mathematical functions.

Syntax:

```
double pow(double x, double y)
decimal pow(decimal x, decimal y)
```

Description: This function calculates the y th power of x , namely, x^y .

Parameters:

- **x**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly

converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

- **y**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If an input value is NULL, NULL is returned.

Examples:

```
select pow(10,2);
```

The following result is returned:

```
+-----+
|_c0  |
+-----+
| 100.0 |
+-----+
```

1.6.6.1.15. RAND

This topic describes the RAND function in mathematical functions and provides examples.

Syntax:

```
double rand(bigint seed)
```

Description: This function returns a random number of the DOUBLE type based on seed. The value ranges from 0 to 1.

Parameters: The seed parameter supports a value of the BIGINT type. It specifies the starting point in generating random numbers. This parameter is optional.

Return value: A value of the DOUBLE type is returned.

Examples:

```
select rand();
select rand(1);
```

1.6.6.1.16. ROUND

This topic describes the ROUND function in mathematical functions and provides examples.

Syntax:

```
double round(double number, [bigint decimal_places])
decimal round(decimal number, [bigint decimal_places])
```

Description: This function returns a number rounded to the specified decimal place.

Parameters:

- **number**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

- `decimal_places`: a constant of the BIGINT type. It specifies the decimal place to which the number is rounded. If the input value is of another data type, an error is returned. If this parameter is not specified, the number is rounded to the ones place. The default value is 0.

 **Note** `decimal_places` supports negative values. A negative value indicates counting from the decimal point to the left, and the decimal part is excluded. If `decimal_places` exceeds the length of the integer part, 0 is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If an input value is NULL, NULL is returned.

Examples:

```
round(125.315) = 125.0
round(125.315, 0) = 125.0
round(125.315, 1) = 125.3
round(125.315, 2) = 125.32
round(125.315, 3) = 125.315
round(-125.315, 2) = -125.32
round(123.345, -2) = 100.0
round(null) = null
round(123.345, 4) = 123.345
round(123.345, -4) = 0.0
```

1.6.6.1.17. SIN

This topic describes the SIN function in mathematical functions.

Syntax:

```
double sin(double number)
decimal sin(decimal number)
```

Description: This function calculates the sine of number, which is a radian value.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.18. SINH

This topic describes the SINH function in mathematical functions.

Syntax:

```
double sinh(double number)
decimal sinh(decimal number)
```

Description: This function calculates the hyperbolic sine of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.19. SQRT

This topic describes the SQRT function in mathematical functions.

Syntax:

```
double sqrt(double number)
decimal sqrt(decimal number)
```

Description: This function calculates the square root of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. It must be greater than 0. If it is less than 0, an error is returned. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.20. TAN

This topic describes the TAN function in mathematical functions.

Syntax:

```
double tan(double number)
decimal tan(decimal number)
```

Description: This function calculates the tangent of number, which is a radian value.

Parameters: number, a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.21. TANH

This topic describes the TANH function in mathematical functions.

Syntax:

```
double tanh(double number)
decimal tanh(decimal number)
```

Description: This function calculates the hyperbolic tangent of number.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted to a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If the input value is NULL, NULL is returned.

1.6.6.1.22. TRUNC

This topic describes the TRUNC function in mathematical functions and provides examples.

Syntax:

```
double trunc(double number[, bigint decimal_places])
decimal trunc(decimal number[, bigint decimal_places])
```

Description: This function truncates the input value of number to a specified number of decimal places.

Parameters:

- number: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into a value of the DOUBLE type before calculation. If the input value is of another data type, an error is returned.
- decimal_places: the decimal place, which is a constant of the BIGINT type. This parameter indicates the position where the number is truncated. If it is of another data type, it is implicitly converted into the BIGINT type. If you do not specify this parameter, the number is truncated to the ones place.

 **Note** decimal_places can be a negative value, which indicates that the number is truncated from the decimal point to the left and the decimal part is left out. If decimal_places exceeds the length of the integer part, 0 is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned. If an input value is NULL, NULL is returned.

 **Note**

- If a value of the DOUBLE type is returned, the return value may not be properly displayed. This issue exists in all systems. For more information, see `trunc(125.815,1)` in the following example.
- The number is filled with zeros from the specified position.

Examples:

```
trunc(125.815) = 125.0
trunc(125.815, 0) = 125.0
trunc(125.815, 1) = 125.800000000000001
trunc(125.815, 2) = 125.81
trunc(125.815, 3) = 125.815
trunc(-125.815, 2) = -125.81
trunc(125.815, -1) = 120.0
trunc(125.815, -2) = 100.0
trunc(125.815, -3) = 0.0
trunc(123.345, 4) = 123.345
trunc(123.345, -4) = 0.0
```

1.6.6.1.23. Additional mathematical functions

1.6.6.1.23.1. Usage notes

This topic provides notes for you to use additional mathematical functions.

MaxCompute 2.0 provides additional mathematical functions. If you want to call additional mathematical functions, you must add the following SET statement before the SQL statement that contains the additional mathematical functions:

```
set odps.sql.type.system.odps2=true;
```

 **Note** You must simultaneously submit and execute the SET statement and the SQL statement that contains the additional mathematical functions.

1.6.6.1.23.2. LOG2

This topic describes the LOG2 function in mathematical functions and provides examples.

Syntax:

```
double log2(double number)
double log2(decimal number)
```

Description: This function calculates the logarithm of number with the base number of 2.

Parameters: number, a value of the DOUBLE or DECIMAL type.

Return value: A value of the DOUBLE type is returned. If the input value is 0 or NULL, NULL is returned.

Examples:

```
log2(null) = null
log2(0) = null
log2(8) = 3.0
```

1.6.6.1.23.3. LOG10

This topic describes the LOG10 function in mathematical functions and provides examples.

Syntax:

```
double log10(double number)
double log10(decimal number)
```

Description: This function calculates the logarithm of number with the base number of 10.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type.

Return value: A value of the DOUBLE type is returned. If the input value is 0 or NULL, NULL is returned.

Examples:

```
log10(null)=null
log10(0)=null
log10(8)=0.9030899869919435
```

1.6.6.1.23.4. BIN

This topic describes the BIN function in mathematical functions and provides examples.

Syntax:

```
string bin(bigint number)
```

Description: This function calculates the binary code of number.

Parameters: The number parameter supports a value of the BIGINT type.

Return value: A value of the STRING type is returned. If the input value is 0, 0 is returned. If the input value is NULL, NULL is returned.

Examples:

```
bin(0) = '0'  
bin(null) = 'null'  
bin(12) = '1100'
```

1.6.6.1.23.5. HEX

This topic describes the HEX function in mathematical functions and provides examples.

Syntax:

```
string hex(bigint number)  
string hex(string number)  
string hex(binary number)
```

Description: This function converts an integer or a string into a hexadecimal number.

Parameters: If number is of the BIGINT type, a hexadecimal number is returned. If number is of the STRING type, a string in hexadecimal format is returned.

Return value: A value of the STRING type is returned. If the input value is 0, 0 is returned. If the input value is NULL, NULL is returned.

Examples:

```
hex(0) = '0'  
hex('abc') = '616263'  
hex(17) = '11'  
hex('17') = '3137'  
hex(null)  
-- An exception occurs and the execution fails.
```

1.6.6.1.23.6. UNHEX

This topic describes the UNHEX function in mathematical functions and provides examples.

Syntax:

```
binary unhex(string number)
```

Description: This function converts a hexadecimal string into a string.

Parameters: The number parameter specifies a hexadecimal string.

Return value: A value of the BINARY type is returned. If the input value is 0, an error is returned. If the input value is NULL, NULL is returned.

Examples:

```
unhex('616263') = 'abc'  
unhex(616263) = 'abc'
```

1.6.6.1.23.7. RADIANS

This topic describes the RADIANS function in mathematical functions and provides examples.

Syntax:

```
double radians(double number)
```

Description: This function converts a degree into a radian value.

Parameters: number, a value of the DOUBLE type.

Return value: A value of the DOUBLE type is returned. If the input value is NULL, NULL is returned.

Examples:

```
radians(90) = 1.5707963267948966  
radians(0) = 0.0  
radians(null) = null
```

1.6.6.1.23.8. DEGREES

This topic describes the DEGREES function in mathematical functions and provides examples.

Syntax:

```
double degrees(double number)  
double degrees(decimal number)
```

Description: This function converts a radian value to a degree.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type.

Return value: A value of the DOUBLE type is returned. If the input value is NULL, NULL is returned.

Examples:

```
degrees(1.5707963267948966) = 90.0  
degrees(0) = 0.0  
degrees(null) = null
```

1.6.6.1.23.9. SIGN

This topic describes the SIGN function in mathematical functions and provides examples.

Syntax:

```
double sign(double number)  
double sign(decimal number)
```

Description: This function returns the sign of an input value.

Parameters: The number parameter supports a value of the DOUBLE or DECIMAL type.

Return value: A value of the DOUBLE type is returned. If the input value is a positive value, 1.0 is returned. If the input value is a negative value, -1.0 is returned. If the input value is 0, 0.0 is returned. If the input value is NULL, NULL is returned.

Examples:

```
sign(-2.5) = -1.0
sign(2.5) = 1.0
sign(0) = 0.0
sign(null) = null
```

1.6.6.1.23.10. E

This topic describes the E function in mathematical functions and provides examples.

Syntax:

```
double e()
```

Description: This function returns the value of e.

Return value: A value of the DOUBLE type is returned.

Examples:

```
e() = 2.718281828459045
```

1.6.6.1.23.11. PI

This topic describes the PI function in mathematical functions and provides examples.

Syntax:

```
double pi()
```

Description: This function calculates the value of π .

Return value: A value of the DOUBLE type is returned.

Examples:

```
pi() = 3.141592653589793
```

1.6.6.1.23.12. FACTORIAL

This topic describes the FACTORIAL function in mathematical functions and provides examples.

Syntax:

```
bigint factorial(int number)
```

Description: This function calculates the factorial of number.

Parameters: number indicates a value of the INT type. The value ranges from 0 to 20.

Return value: A value of the BIGINT type is returned. If the input value is 0, 1 is returned. If the input value is NULL or a value that does not fall into the range from 0 to 20, NULL is returned.

Examples:

```
factorial(5) = 120 --5! = 5*4*3*2*1 = 120
```

1.6.6.1.23.13. CBRT

This topic describes the CBRT function in mathematical functions and provides examples.

Syntax:

```
double cbrt(double number)
```

Description: This function calculates the cube root of number.

Parameters: The number parameter supports a value of the DOUBLE type.

Return value: A value of the DOUBLE type is returned. If the input value is NULL, NULL is returned.

Examples:

```
cbrt(8) = 2  
cbrt(null) = null
```

1.6.6.1.23.14. SHIFTLEFT

This topic describes the SHIFTLEFT function in mathematical functions and provides examples.

Syntax:

```
int shiftleft(tinyint|smallint|int number1, int number2)  
bigint shiftleft(bigint number1, int number2)
```

Description: This function shifts a value left by a specific number of places (<<).

Parameters:

- number1: an integer of the TINYINT, SMALLINT, INT, or BIGINT type.
- number2: an integer of the INT type.

Return value: A value of the INT or BIGINT type is returned.

Examples:

```
shiftleft(1,2)=4  
-- Shift the binary value of 1 two places to the left (1<<2, 0001 shifted to 0100).  
shiftleft(4,3)=32  
-- Shift the binary value of 4 three places to the left (4<<3, 0100 shifted to 100000).
```

1.6.6.1.23.15. SHIFTRIGHT

This topic describes the SHIFTRIGHT function in mathematical functions and provides examples.

Syntax:

```
int shiftright(tinyint|smallint|int number1, int number2)  
bigint shiftright(bigint number1, int number2)
```

Description: This function shifts a value right by a specific number of places (>>).

Parameters:

- number1: an integer of the TINYINT, SMALLINT, INT, or BIGINT type.
- number2: an integer of the INT type.

Return value: A value of the INT or BIGINT type is returned.

Examples:

```
shiftright(4,2) = 1
-- Shift the binary value of 4 two places to the right (4>>2, 0100 shifted to 0001).
shiftright(32,3) = 4
-- Shift the binary value of 32 three places to the right (32>>3, 100000 shifted to 0100).
```

1.6.6.1.23.16. SHIFTRIGHTUNSIGNED

This topic describes the SHIFTRIGHTUNSIGNED function in mathematical functions and provides examples.

Syntax:

```
int shiftrightunsigned(tinyint|smallint|int number1, int number2)
bigint shiftrightunsigned(bigint number1, int number2)
```

Description: This function shifts an unsigned value right by a specific number of places (>>>).

Parameters:

- number1: an integer of the TINYINT, SMALLINT, INT, or BIGINT type.
- number2: an integer of the INT type.

Return value: A value of the INT or BIGINT type is returned.

Examples:

```
shiftrightunsigned(8,2) = 2
-- Shift the binary unsigned value of 8 two places to the right (8>>>2,1000 shifted to be 0010).
shiftrightunsigned(-14,2) = 1073741820
-- Shift the binary value of -14 two places to the right (-14>>>2, 11111111 11111111 11111111 11110010 shifted to be 00111111 11111111 11111111 11111100).
```

1.6.6.1.23.17. FORMAT_NUMBER

This topic describes the FORMAT_NUMBER function in mathematical functions and provides examples.

Syntax:

```
string format_number(float|double|decimal expr1, expr2)
```

Description: This function converts a number to a string in the specified format.

Parameters:

- expr1: a numeric expression that you want to format.
- expr2: the number of decimal places. It can be of the INT type. It can also be expressed in the format of `#,###,###.##`.

 Note

- If expr2 is greater than 0, the value is rounded to the specified place after the decimal point.
- If expr2 is equal to 0, the value has no decimal point or fractional part.
- If expr2 is less than 0 or greater than 340, an error is returned.

Return value: A value of the STRING type is returned.

Examples:

```
select format_number(5.230134523424545456,3);  
-- The value 5.230 is returned.  
select format_number(12332.123456, '#,###,###,###.###');  
-- The value 12,332.123 is returned.
```

1.6.6.1.23.18. WIDTH_BUCKET

This topic describes the WIDTH_BUCKET function in mathematical functions and provides examples.

Syntax:

```
WIDTH_BUCKET(NUMERIC expr, NUMERIC min_value, NUMERIC max_value, INT num_buckets)
```

Description: This function specifies the number of buckets and the minimum and maximum values of the acceptable range for a bucket. It allows you to construct equi-width buckets, in which the bucket range is divided into intervals that have an identical size. It returns the ID of the bucket into which the value of a specific expression falls. This function supports the following data types: DECIMAL(precision,scale) in the MaxCompute V2.0 data type edition, BIGINT, INT, FLOAT, DOUBLE, and DECIMAL.

Parameters:

- expr: the expression for which you want to identify the matching bucket ID.
- min_value: the minimum value of the acceptable range for the bucket.
- max_value: the maximum value of the acceptable range for the bucket. The value must be greater than min_value.
- num_buckets: the number of buckets. The value must be greater than 0.

Return value: A value of the BIGINT type is returned. The value ranges from 0 to num_buckets plus 1. If the value of expr is less than min_value, 0 is returned. If the value of expr is greater than max_value, the value of num_buckets plus 1 is returned. If the value of expr is NULL, NULL is returned. In other cases, the ID of the bucket into which the value falls is returned. The bucket ID is named based on the following formula: $\text{Bucket ID} = \text{FLOOR}[\text{num_buckets} \times (\text{expr} - \text{min_value}) / (\text{max_value} - \text{min_value}) + 1]$.

Examples:

```
SELECT key,value,WIDTH_BUCKET(value,100,500,5) as value_group
FROM VALUES
(1,99),
(2,100),
(3,199),
(4,200),
(5,499),
(6,500),
(7,501),
(8,NULL)
AS t(key,value);
```

The following result is returned:

```
+-----+-----+-----+
| key | value | value_group |
+-----+-----+-----+
| 1 | 99 | 0 |
| 2 | 100 | 1 |
| 3 | 199 | 2 |
| 4 | 200 | 2 |
| 5 | 499 | 5 |
| 6 | 500 | 6 |
| 7 | 501 | 6 |
| 8 | \N | \N |
+-----+-----+-----+
```

1.6.6.2. String functions

1.6.6.2.1. CHAR_MATCHCOUNT

This topic describes the CHAR_MATCHCOUNT function in string functions and provides examples.

Syntax:

```
bigint char_matchcount(string str1, string str2)
```

Description: This function returns the number of characters that belong to str1 and appear in str2.

Parameters: The str1 and str2 parameters support a value of the STRING type. The values must be valid UTF-8 strings. If invalid characters are found during the comparison of the two strings, a negative value is returned.

Return value: A value of the BIGINT type is returned. If an input value is NULL, NULL is returned.

Examples:

```
char_matchcount('abd', 'aabc') = 2
-- The characters a and b of str1 appear in str2.
```

1.6.6.2.2. CHR

This topic describes the CHR function in string functions.

Syntax:

```
string chr(bigint ascii)
```

Description: This function converts specified an ASCII code into the required character.

Parameters: `ascii` indicates the ASCII code in the BIGINT type. If the input value is of the STRING, DOUBLE, or DECIMAL type, it is implicitly converted into a value of the BIGINT type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. The input value ranges from 0 to 255. If the input value does not fall into this range, an error is returned. If an input value is NULL, NULL is returned.

1.6.6.2.3. CONCAT

This topic describes the CONCAT function in string functions and provides examples.

Syntax:

```
string concat(string a, string b...)
```

Description: This function concatenates all specified strings.

Parameters: `a` and `b`, values of the STRING type. If an input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If an input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the parameters are not specified or an input value is NULL, NULL is returned.

Examples:

```
concat('ab', 'c') = 'abc'  
concat() = null  
concat('a', null, 'b') = null
```

1.6.6.2.4. INSTR

This topic describes the INSTR function in string functions and provides examples.

Syntax:

```
bigint instr(string str1, string str2[, bigint start_position[, bigint nth_appearance]])
```

Description: This function determines the position of substring `str2` in string `str1`.

Parameters:

- `str1`: a value of the STRING type, which indicates the string in which you search for the substring. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.
- `str2`: a value of the STRING type, which indicates the substring you want to search for. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.
- `start_position`: a value of the BIGINT type. If the input value is of another data type, an error is returned. It indicates the character in `str1` from which the search starts. The default value is 1, which indicates that the search starts from the first character.
- `nth_appearance`: a value of the BIGINT type. If the value is greater than 0, it indicates the position where

the substring matches the string for the nth_appearance time. If the value is of another data type or less than or equal to 0, an error is returned.

Return value: A value of the BIGINT type is returned. If str2 is not matched in str1, 0 is returned. If an input value is NULL, NULL is returned. If str2 is an empty string, the substring always matches the string.

Examples:

```
instr('Tech on the net', 'e') = 2
instr('Tech on the net', 'e', 1, 1) = 2
instr('Tech on the net', 'e', 1, 2) = 11
instr('Tech on the net', 'e', 1, 3) = 14
```

1.6.6.2.5. IS_ENCODING

This topic describes the IS_ENCODING function in string functions and provides examples.

Syntax:

```
boolean is_encoding(string str, string from_encoding, string to_encoding)
```

Description: This function determines whether the input string str can be converted from the character set specified by from_encoding to the character set specified by to_encoding. It can be used to determine whether the input string is garbled. Generally, from_encoding is set to utf-8, and to_encoding is set to gbk.

Parameters:

- str: a value of the STRING type. An empty string can belong to any character set.
- from_encoding and to_encoding: a value of the STRING type. The two parameters specify the source and destination character sets. If an input value is NULL, NULL is returned.

Return value: A value of the BOOLEAN type is returned. If str can be converted, true is returned. Otherwise, false is returned.

Examples:

```
is_encoding('测试', 'utf-8', 'gbk') = true
is_encoding('測試', 'utf-8', 'gbk') = true
-- The two traditional Chinese characters are valid in the GBK character set.
is_encoding('測試', 'utf-8', 'gb2312') = false
-- The two traditional Chinese characters are invalid in the GB2312 character set.
```

1.6.6.2.6. KEYVALUE

This topic describes the KEYVALUE function in string functions and provides examples.

Syntax:

```
KEYVALUE(STRING srcStr, STRING split1, STRING split2, STRING key)
KEYVALUE(STRING srcStr, STRING key) //split1 = ";", split2 = ":"
```

Description: This function splits the source string srcStr into key-value pairs by split1, separates key-value pairs by split2, and then returns the value of the specified key.

Parameters:

- srcStr: the source string that you want to split.

- **key**: a value of the STRING type. After you split the source string by **split1** and **split2**, the value of the specified key is returned.
- **split1** and **split2**: the delimiters that are used to split strings. You must split the source string by using the specified delimiters. If you do not specify these two parameters, the default value of **split1** is a semicolon (;) and that of **split2** is a colon (:). If multiple key-value pairs are obtained after the source string is split by **split1** and multiple delimiters specified by **split2** are used to split the pairs, the returned result is undefined.

Return value: A value of the STRING type is returned. If the value of **split1** or **split2** is NULL, NULL is returned. If the value of **srcStr** or **key** is NULL, or **key** is not matched, NULL is returned. If multiple key-value pairs match the **key**, the value that corresponds to the first matched key is returned.

Examples:

Example 1:

```
KEYVALUE('0:1\;1:2', 1) = '2'
```

The source string is "0:1\;1:2". **split1** and **split2** use their default values. The default value of **split1** is a semicolon (;) and that of **split2** is a colon (:). After the source string is split by **split1**, the key-value pairs of 0:1\,1:2 are generated. After the key-value pairs are split by **split2**, the key-value pairs change to the following form:

```
0 1/  
1 2  
-- The value 2 that corresponds to key 1 is returned.
```

Example 2:

```
KEYVALUE("\;decreaseStore:1\;xcard:1\;isB2C:1\;tf:21910\;cart:1\;shipping:2\;pf:0\;market:shoes\;instPayAmount:0\;  
","\;",";", "tf") = "21910" value:21910
```

The source string is

```
\;decreaseStore:1\;xcard:1\;isB2C:1\;tf:21910\;cart:1\;shipping:2\;pf:0\;market:shoes\;instPayAmount:0\;
```

After the source string is split by \; specified by **split1**, the following key-value pairs are generated:

```
decreaseStore:1, xcard:1, isB2C:1, tf:21910, cart:1, shipping:2, pf:0, market:shoes, instPayAmount:0
```

After the key-value pairs are split by a colon (:) specified by **split2**, the key-value pairs change to the following form:

```
decreaseStore 1  
xcard 1  
isB2C 1  
tf 21910  
cart 1  
shipping 2  
pf 0  
market shoes  
instPayAmount 0  
-- The value 21910 that corresponds to key tf is returned.
```

1.6.6.2.7. LENGTH

This topic describes the LENGTH function in string functions and provides examples.

Syntax:

```
bigint length(string str)
```

Description: This function returns the length of the string str.

Parameters: The str parameter supports a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If the input value is NULL, NULL is returned. If the input value is not encoded in UTF-8, -1 is returned.

Examples:

```
length('china') = 5
```

1.6.6.2.8. LENGTHB

This topic describes the LENGTHB function in string functions and provides examples.

Syntax:

```
bigint lengthb(string str)
```

Description: This function calculates the length of str in bytes.

Parameters: str, a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If the input value is NULL, NULL is returned.

Examples:

```
lengthb('hi! China') = 10
```

1.6.6.2.9. MD5

This topic describes the MD5 function in string functions.

Syntax:

```
string md5(string value)
```

Description: This function calculates the MD5 value of value.

Parameters: value, a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

1.6.6.2.10. PARSE_URL

This topic describes the PARSE_URL function in string functions and provides examples.

Syntax:

```
string parse_url(string url, string part[,string key])
```

Description: This function parses url and extracts information by key.

Parameters:

- url: the URL to parse. If url is NULL, NULL is returned. If url is an invalid URL, an error is returned.
- part: a value of the STRING type. Valid values: HOST, PATH, QUERY, REF, PROTOCOL, AUTHORITY, FILE, and USERINFO. The input value is not case-sensitive. If the input value is invalid, an error is returned.
- key: If part is QUERY, the value of key is extracted from the query string in URL and is returned. Otherwise, key is omitted.

Return value: A value of the STRING type is returned.

Examples:

```
url = file://username:password@example.com:8042/over/there/index.dtb? type=animal&name=narwhal#nose
parse_url('url', 'HOST') = "example.com"
parse_url('url', 'PATH') = "/over/there/index.dtb"
parse_url('url', 'QUERY') = "type=animal&name=narwhal"
parse_url('url', 'QUERY', 'name') = "narwhal"
parse_url('url', 'REF') = "nose"
parse_url('url', 'PROTOCOL') = "file"
parse_url('url', 'AUTHORITY') = "username:password@example.com:8042"
parse_url('url', 'FILE') = "/over/there/index.dtb? type=animal&name=narwhal"
parse_url('url', 'USERINFO') = "username:password"
```

1.6.6.2.11. REGEXP_EXTRACT

This topic describes the REGEXP_EXTRACT function in string functions and provides examples.

Syntax:

```
string regexp_extract(string source, string pattern[, bigint occurrence])
```

Description: This function splits source based on the regular expression specified by pattern and returns the characters in group at the nth occurrence, where n is specified by occurrence.

Parameters:

- source: a value of the STRING type. This parameter indicates the string that you want to split.
- pattern: a constant of the STRING type. If pattern is an empty string or group is not specified in pattern, an error is returned.
- occurrence: a constant of the BIGINT type, which must be greater than or equal to 0. If the input value is of another data type or is smaller than 0, an error is returned. If you do not specify this parameter, the default value 1 is used. The value 1 indicates that characters in the first group are returned. If you set occurrence to 0, all substrings that match the regular expression specified by pattern are returned.

 **Note** Data is encoded by using UTF-8.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```

regexp_extract('foothebar', 'foo(. *?)( bar)', 1) = the
regexp_extract('foothebar', 'foo(. *?)( bar)', 2) = bar
regexp_extract('foothebar', 'foo(. *?)( bar)', 0) = foothebar
regexp_extract('8d99d8', '8d(\\d+)d8') = 99
-- If the regular expression is submitted on the MaxCompute client, use two backslashes (\\) as the escape character
s.
regexp_extract('foothebar', 'foothebar')
-- An error is returned because group is not specified in pattern.

```

1.6.6.2.12. REGEXP_INSTR

This topic describes the REGEXP_INSTR function in string functions and provides examples.

Syntax:

```

bigint regexp_instr(string source, string pattern[, bigint start_position[, bigint nth_occurrence[, bigint return_option]])

```

Description: This function returns the start or end position of the substring that matches pattern at the nth occurrence, where n is specified by nth_occurrence, in the source string from the start position specified by start_position.

Parameters:

- source: a value of the STRING type. This parameter indicates the string that you want to search.
- pattern: a constant of the STRING type. If pattern is an empty string, an error is returned.
- start_position: a constant of the BIGINT type. This parameter indicates the start position for the search. If you do not specify this parameter, the default value 1 is used. If the value is of another data type or less than or equal to 0, an error is returned.
- nth_occurrence: a constant of the BIGINT type. If you do not specify this parameter, the default value 1 is used, indicating the position where a substring matches pattern in the search for the first time. If the value is of another data type or less than or equal to 0, an error is returned.
- return_option: a constant of the BIGINT type. Valid values are 0 and 1. If the value is of another data type or invalid, an error is returned. The value 0 indicates that the start position of the matched substring is returned. The value 1 indicates that the end position of the matched substring is returned.

Return value: A value of the BIGINT type is returned. This parameter indicates the start or end position of the matched substring returned in the source string based on the type specified by return_option. If an input value is NULL, NULL is returned.

Examples:

```

regexp_instr("i love www.taobao.com", "o[[:alpha:]]{1}", 3, 2) = 14

```

1.6.6.2.13. REGEXP_SUBSTR

This topic describes the REGEXP_SUBSTR function in string functions and provides examples.

Syntax:

```

string regexp_substr(string source, string pattern[, bigint start_position[, bigint nth_occurrence]])

```

Description: This function returns the substring that matches a given pattern at the nth occurrence, where n is specified by nth_occurrence, in the source string from the start position specified by start_position.

Parameters:

- **source**: a value of the STRING type. It specifies the string that contains the substring to return.
- **pattern**: a constant of the STRING type. It specifies the pattern used to match the substring. If the pattern value is NULL, an error is returned.
- **start_position**: a constant of the BIGINT type. It must be greater than 0. If it is of another data type or is less than or equal to 0, the function returns an error. If start_position is not specified, the default value 1 is used. In this case, the function starts matching from the first character of the source string.
- **nth_occurrence**: a constant of the BIGINT type. It must be greater than 0. If it is of another data type or is less than or equal to 0, an error is returned. If it is not specified, the default value 1 is used. In this case, the function returns the substring that first matches the specified pattern.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned. If no substrings match the specified pattern, NULL is returned.

Examples:

```
regexp_substr("I love aliyun very much", "a[[:alpha:]]{5}") = "aliyun"  
regexp_substr("I have 2 apples and 100 bucks!", '[:blank:][:alnum:]*', 1, 1) = " have"  
regexp_substr("I have 2 apples and 100 bucks!", '[:blank:][:alnum:]*', 1, 2) = " 2"
```

1.6.6.2.14. REGEXP_COUNT

This topic describes the REGEXP_COUNT function in string functions and provides examples.

Syntax:

```
bigint regexp_count(string source, string pattern[, bigint start_position])
```

Description: This function returns the number of times a substring matches a given pattern in the source string from the start position specified by start_position.

Parameters:

- **source**: a value of the STRING type. It specifies the string that contains the substring to match. If the input value is of another data type, an error is returned.
- **pattern**: a constant of the STRING type. It specifies the pattern used to match the substring. If the pattern value is an empty string or is of another data type, an error is returned.
- **start_position**: a constant of the BIGINT type. It must be greater than 0. If it is of another data type or is less than or equal to 0, the function returns an error. If start_position is not specified, the default value 1 is used. In this case, the function starts matching from the first character of the source string.

Return value: A value of the BIGINT type is returned. If an input value is NULL, NULL is returned. If no substrings match the specified pattern, 0 is returned.

Examples:

```
regexp_count('abababc', 'a.c') = 1  
regexp_count('abcde', '[:alpha:]{2}', 3) = 1
```

1.6.6.2.15. REGEXP_REPLACE

This topic describes the REGEXP_REPLACE function in string functions and provides examples.

Syntax:

```
string regexp_replace(string source, string pattern, string replace_string[, bigint occurrence])
```

Description: This function substitutes the string specified by `replace_string` for the substring that matches a given pattern at the `n`th occurrence, where `n` is specified by `occurrence`, in the source string, and returns the result.

Parameters:

- `source`: a value of the `STRING` type. It specifies the string that contains the substring to substitute.
- `pattern`: a constant of the `STRING` type. It specifies the pattern used to match the substring. If the value of `pattern` is `NULL`, an error is returned.
- `replace_string`: a value of the `STRING` type. It specifies the string to substitute for the substring that matches `pattern`.
- `occurrence`: a constant of the `BIGINT` type. It specifies the number of times at which the substring matches the pattern and is substituted with `replace_string`. It must be greater than or equal to 0. If the input value is 0, all matched substrings are substituted. If the input value is less than 0 or is not of the `BIGINT` type, an error is returned. Default value: 0.

 **Note** The action of referencing a character class that does not exist is undefined.

Return value: A value of the `STRING` type is returned. If the function references a character class that does not exist, substitution is not performed. If `replace_string` is `NULL` and substrings match `pattern`, `NULL` is returned. If `replace_string` is `NULL` and no substrings match `pattern`, the source string is returned. If an input value is `NULL`, `NULL` is returned.

Examples:

```
regexp_replace("123.456.7890", "[[:digit:]]{3}\\.([[:digit:]]{3})\\.([[:digit:]]{4})", "(\\1)\\2-\\3", 0) = "(123)456-7890"
regexp_replace("abcd", "(.)", "\\1 ", 0) = "a b c d "
regexp_replace("abcd", "(.)", "\\1 ", 1) = "a bcd"
regexp_replace("abcd", "(.)", "\\2", 1) = "abcd"
-- Only one character class is defined in pattern. The second referenced character class does not exist.
-- Try to avoid this action. The result of referencing a non-existent group is undefined.
regexp_replace("abcd", "(.*)\\.($)", "\\2", 0) = "d"
regexp_replace("abcd", "a", "\\1", 0) = "bcd"
-- No character class is defined in pattern. \\1 references a non-existent character class.
-- Try to avoid this action. The result of referencing a non-existent group is undefined.
```

1.6.6.2.16. SPLIT_PART

This topic describes the `SPLIT_PART` function in string functions and provides examples.

Syntax:

```
string split_part(string str, string separator, bigint start[, bigint end])
```

Description: This function uses a delimiter specified by `separator` to split `str`, and returns a substring that starts from the character specified by `start` and ends with the character specified by `end`.

Parameters:

- `str`: a value of the `STRING` type, which indicates the string that you want to split. If the input value is of the `BIGINT`, `DOUBLE`, `DECIMAL`, or `DATETIME` type, it is implicitly converted into a value of the `STRING` type before calculation. If the input value is of another data type, an error is returned.
- `separator`: a constant of the `STRING` type, which indicates the delimiter used to split a string. It can be a

character or string. If it is of another data type, an error is returned.

- **start**: a constant of the BIGINT type, which must be greater than 0. If the value is not a constant or is of another data type, an error is returned. It indicates the start number of the segment to be returned. The number starts from 1. If end is not specified, the segment specified by start is returned.
- **end**: a constant of the BIGINT type. It must be greater than or equal to the value of start. Otherwise, an error is returned. It indicates the end number of the segment to be returned. If the value is not a constant or is of another data type, an error is returned. If this parameter is not specified, the last segment is returned.

Return value: A value of the STRING type is returned.

- If you set start to a value greater than the number of segments, an empty string is returned.
- If separator is absent in str and start is set to 1, the entire str is returned.
- If str is an empty string, an empty string is returned.
- If separator is an empty string, str is returned.
- If you set end to a value greater than the number of segments, all segments are returned.
- If an input value is NULL, NULL is returned.

Examples:

```
split_part('a,b,c,d', ',', 1) = 'a'  
split_part('a,b,c,d', ',', 1, 2) = 'a,b'  
split_part('a,b,c,d', ',', 10) = ''
```

1.6.6.2.17. SUBSTR

This topic describes the SUBSTR function in string functions and provides examples.

Syntax:

```
string substr(string str, bigint start_position[, bigint length])
```

Description: This function returns a substring that starts from start_position in str and has a length specified by length.

Parameters:

- **str**: a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.
- **start_position**: a value of the BIGINT type. The function starts counting from the first character. If start_position is a negative value, the start position is counted backward from the last character of the string. The value -1 indicates the last character. If the input value is of another data type, an error is returned.
- **length**: a value of the BIGINT type. It specifies the length of the substring. The input value must be greater than 0. If the input value is of another data type or is less than or equal to 0, an error is returned.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

 **Note** If length is not specified, this function returns a substring that starts from start_position in str to the end of str.

Examples:

```
substr("abc", 2) = "bc"  
substr("abc", 2, 1) = "b"  
substr("abc", -2, 2) = "bc"  
substr("abc", -3) = "abc"
```

1.6.6.2.18. TOWER

This topic describes the TOWER function in string functions and provides examples.

Syntax:

```
string tower(string source)
```

Description: This function converts the string source to lowercase letters.

Parameters: The source parameter supports a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
tower("aBcd") = "abcd"  
tower("HAHACd") = "hahacd"
```

1.6.6.2.19. TUPPER

This topic describes the TUPPER function in string functions and provides examples.

Syntax:

```
string tupper(string source)
```

Description: This function converts source into uppercase letters.

Parameters: source, a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
tupper("aBcd") = "ABCD"  
tupper("HahaCd") = "HAHACD"
```

1.6.6.2.20. TO_CHAR

This topic describes the TO_CHAR function in string functions and provides examples.

Syntax:

```
string to_char(boolean value)
string to_char(bigint value)
string to_char(double value)
string to_char(decimal value)
```

Description: This function converts data of the BOOLEAN, BIGINT, DECIMAL, or DOUBLE type into the STRING type.

Parameters: value, a value of the BOOLEAN, BIGINT, DECIMAL, or DOUBLE type. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
to_char(123) = '123'
to_char(true) = 'TRUE'
to_char(1.23) = '1.23'
to_char(null) = 'null'
```

1.6.6.2.21. TRIM

This topic describes the TRIM function in string functions.

Syntax:

```
string trim(string str)
```

Description: This function eliminates the spaces on the left and right sides of the string str.

Parameters: The str parameter supports a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, an error is returned.

1.6.6.2.22. LTRIM

This topic describes the LTRIM function in string functions and provides examples.

Syntax:

```
string ltrim(string str)
```

Description: This function removes the spaces on the left side of str.

Parameters: str, a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
select ltrim(' abc ');
-- The following result is returned:
+-----+
| _c0 |
+-----+
| abc |
+-----+
```

1.6.6.2.23. RTRIM

This topic describes the RTRIM function in string functions and provides examples.

Syntax:

```
string rtrim(string str)
```

Description: This function eliminates the spaces on the right side of the string str.

Parameters: The str parameter supports a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
select rtrim('a abc ');
-- The following result is returned:
+-----+
| _c0 |
+-----+
| a abc |
+-----+
```

1.6.6.2.24. REVERSE

This topic describes the REVERSE function in string functions and provides examples.

Syntax:

```
string reverse(string str)
```

Description: This function returns a string in reverse order.

Parameters: The str parameter supports a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, an error is returned.

Examples:

```
select reverse('abcdfg');
-- The following result is returned:
+-----+
| _c0 |
+-----+
| gfdecba |
+-----+
```

1.6.6.2.25. SPACE

This topic describes the SPACE function in string functions and provides examples.

Syntax:

```
string space(bigint n)
```

Description: This function returns a string of a length `h` specified by `n`.

Parameters: The `n` parameter supports a value of the BIGINT type. The value cannot exceed 2 MB. If this parameter is left empty, an error is returned.

Return value: A value of the STRING type is returned.

Examples:

```
select length(space(10));
-- The value 10 is returned.
select space(400000000000);
-- An error is returned because the length exceeds 2 MB.
```

1.6.6.2.26. REPEAT

This topic describes the REPEAT function in string functions and provides examples.

Syntax:

```
string repeat(string str, bigint n)
```

Description: This function returns `n` duplicates of the string `str`.

Parameters:

- `str`: a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.
- `n`: a value of the BIGINT type. The returned string cannot exceed 2 MB in length. If this parameter is left empty, an error is returned.

Return value: A value of the STRING type is returned.

Examples:

```
select repeat('abc',5);
-- abcabcabcabcabc is returned.
```

1.6.6.2.27. ASCII

This topic describes the ASCII function in string functions and provides examples.

Syntax:

```
bigint ascii(string str)
```

Description: This function returns the ASCII code of the first character in the string str.

Parameters: The str parameter supports a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned.

Examples:

```
select ascii('abcde');  
-- The value 97 is returned.
```

1.6.6.2.28. Additional string functions

1.6.6.2.28.1. Usage notes

This topic provides notes for you to use additional string functions.

MaxCompute V2.0 provides additional string functions. You must add the following flag before the SQL statement that contains the additional string functions:

```
set odps.sql.type.system.odps2=true;
```

 **Note** You must submit the flag along with the SQL statement that contains the additional string functions for execution.

1.6.6.2.28.2. CONCAT_WS

This topic describes the CONCAT_WS function in string functions and provides examples.

Syntax:

```
string concat_ws(string SEP, string a, string b...)  
string concat_ws(string SEP, array)
```

Description: This function concatenates all input strings in an array by using a specified delimiter.

Parameters:

- SEP: the delimiter of the STRING type. If this parameter is not specified, an error is returned.
- a, b...: a value of the STRING type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If no input values are specified or an input value is NULL, NULL is returned.

Examples:

```
concat_ws(':', 'name', 'hanmeimei')='name:hanmeimei'  
concat_ws(':', 'avg', null, '34')=null
```

1.6.6.2.28.3. LPAD

This topic describes the LPAD function in string functions and provides examples.

Syntax:

```
string lpad(string a, int len, string b)
```

Description: This function pads the left side of a with b based on the length specified by len.

Parameters:

- len: a value of the INT type.
- a: a value of the STRING type.
- b: a value of the STRING type.

Return value: A value of the STRING type is returned. If len is smaller than the number of characters in a, a is truncated from the left to obtain a string with the number of characters specified by len. If len is 0, no value is returned.

Examples:

```
lpad('abcdefgh', 10, '12')='12abcdefgh'  
lpad('abcdefgh', 5, '12')='abcde'  
lpad('abcdefgh', 0, '12')  
-- No value is returned.
```

1.6.6.2.28.4. RPAD

This topic describes the RPAD function in string functions and provides examples.

Syntax:

```
string rpad(string a, int len, string b)
```

Description: This function pads the right side of string a with string b until the new padded string has len characters.

Parameters:

- len: a value of the INT type.
- a: a value of the STRING type.
- b: a value of the STRING type.

Return value: A value of the STRING type is returned. If len is smaller than the number of characters in a, a is truncated from the left to obtain a string with len characters, and the obtained string is returned. If len is 0, NULL is returned.

Examples:

```
rpad('abcdefgh',10,'12')='abcdefgh12'  
rpad('abcdefgh',5,'12')='abcde'  
rpad('abcdefgh',0,'12')  
-- NULL is returned.
```

1.6.6.2.28.5. REPLACE

This topic describes the REPLACE function in string functions and provides examples.

Syntax:

```
string replace(string a, string OLD, string NEW)
```

Description: This function substitutes the string NEW for the part of string a that is exactly the same as the string OLD, and returns the string a.

Parameters: All the parameters support a value of the STRING type.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```
replace('ababab','abab','12')='12ab'  
replace('ababab','cdf','123')='ababab'  
replace('123abab456ab',null,'abab')=null
```

1.6.6.2.28.6. SOUNDEX

This topic describes the SOUNDEX function in string functions and provides examples.

Syntax:

```
string soundex(string a)
```

Description: This function converts a normal string to a string of the soundex type.

Parameters: The a parameter supports a value of the STRING type.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

Examples:

```
soundex('hello') = 'H400'
```

1.6.6.2.28.7. SUBSTRING_INDEX

This topic describes the SUBSTRING_INDEX function in string functions and provides examples.

Syntax:

```
string substring_index(string a, string SEP, int count)
```

Description: This function truncates the string a to a substring from the first character to the nth delimiter, where n is specified by count. If count is a positive value, the string is truncated from left to right. Otherwise, the string is truncated from right to left.

Parameters:

- a: a value of the STRING type.
- SEP: a value of the STRING type.
- count: a value of the INT type.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```
substring_index('https://help.aliyun.com', '.', 2)='https://help.aliyun'  
substring_index('https://help.aliyun.com', '.', -2)='aliyun.com'  
substring_index('https://help.aliyun.com', null, 2)=null
```

1.6.6.2.28.8. TRANSLATE

This topic describes the TRANSLATE function in string functions and provides examples.

Syntax:

```
string translate(string|varchar str1, string|varchar str2, string|varchar str3)
```

Description: This function replaces the common substring of str1 and str2 with str3.

Parameters: The str parameter supports a value of the STRING or VARCHAR type. If the input value is of the BIGINT, DECIMAL, DOUBLE, or DATETIME type, it is implicitly converted to a value of the STRING type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```
translate('MaxComputer', 'puter', 'pute')='MaxCompute'  
translate('aaa', 'b', 'c')='aaa'  
translate('MaxComputer', 'puter', null)=null
```

1.6.6.2.28.9. URL_ENCODE

This topic describes the URL_ENCODE function in string functions and provides examples.

Syntax:

```
string url_encode(string input[, string encoding])
```

Description: This function encodes the input string in the application/x-www-form-urlencoded MIME format and returns the encoded string.

- All letters remain unchanged.
- Periods (.), hyphens (-), asterisks (*), and underscores (_) remain unchanged.
- Spaces are converted to plus signs (+).
- Other characters are converted to byte values based on encoding. Each byte value is then represented in the format of %xy, where xy is the hexadecimal representation of the character value.

Parameters:

- input: the string to encode.

- encoding: the encoding format. GBK and UTF-8 are supported. If you do not specify this parameter, the default value UTF-8 is used.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```
URL_ENCODE('Example for URL_ENCODE:// (fdsf)') = "%E7%A4%BA%E4%BE%8Bfor+URL_ENCODE%3A%2F%2F+%28fdsf%29"
URL_ENCODE('Example for URL_ENCODE:// dsf(fasfs)', 'GBK') = "Example+for+URL_ENCODE+%3A%2F%2F+dsf%28fasfs%29"
```

1.6.6.2.28.10. URL_DECODE

This topic describes the URL_DECODE function in string functions and provides examples.

Syntax:

```
string url_decode(string input[, string encoding])
```

Description: This function converts an input string from the application/x-www-form-urlencoded MIME format into a normal string. This is the inverse function of URL_ENCODE.

- All letters remain unchanged.
- Periods (.), hyphens (-), asterisks (*), and underscores (_) remain unchanged.
- Each plus sign (+) is converted into a space.
- The function first performs decoding based on the percent sign (%). The sequences that are in the %xy format are converted into byte values. Then, the function performs decoding based on the encoding format specified by the second parameter. Consecutive byte values are decoded to the required strings based on the format specified by encoding.
- Other characters remain unchanged.
- The return value of the function is a string encoded in UTF-8.

Parameters:

- input: the string that you want to decode.
- encoding: the specified encoding format. Valid values include GBK and UTF-8. If you do not specify this parameter, the default value UTF-8 is used.

Return value: A value of the STRING type is returned. If an input value is NULL, NULL is returned.

Examples:

```
URL_DECODE('%E7%A4%BA%E4%BE%8Bfor+URL_DECODE%3A%2F%2F+%28fdsf%29')= "Example for URL_DECODE:// (fdsf)"
URL_DECODE('Example+for+URL_DECODE+%3A%2F%2F+dsf%28fasfs%29', 'GBK') = "Example for URL_DECODE:// dsf(fasfs)"
```

1.6.6.2.28.11. JSON_TUPLE

This topic describes the JSON_TUPLE function in string functions and provides examples.

Syntax:

```
STRING JSON_TUPLE(STRING json,STRING key1,STRING key2,...)
```

Description: This function extracts specific strings from a standard JSON string based on a set of input keys, such as key1 and key2.

Parameters:

- json: a value of the STRING type. It specifies a standard JSON string.
- key: a value of the STRING type. It describes the JSON path. You can enter multiple keys at a time. A key cannot start with a dollar sign (\$).

Return value: A value of the STRING type is returned.

- If the JSON parameter is empty or invalid, NULL is returned.
- If the key parameter is empty or invalid, NULL is returned. If the key value does not exist in the JSON string, it is considered invalid.
- If the JSON parameter is valid and the key value exists in the JSON string, the required string is returned.

 Note

- This function can parse JSON data that contains Chinese characters.
- This function can parse nested JSON data.
- This function can parse JSON data that contains nested arrays.
- This function parses a JSON string in the same way as the GET_JSON_OBJECT function for which `set odps.sql.udf.getjsonobj.new=true;` is added. To parse a JSON string multiple times, you must call the GET_JSON_OBJECT function multiple times. However, the JSON_TUPLE function allows you to enter multiple keys at a time and parse the JSON string only once. This improves parsing efficiency.
- JSON_TUPLE is a user-defined table-valued function (UDTF). To select other columns, you can use JSON_TUPLE with LATERAL VIEW.

Examples:

Prepare a school table that contains the following data:

```
Table: school
+-----+-----+
| Id   | json |
+-----+-----+
| 1   | {
    "School name": "Hupan college",
    "Location": "Hangzhou",
    "SchoolRank": "00",
    "Class1": {
      "Student": [
        {
          "studentId": 1,
          "scoreRankIn3Year": [1, 2, [3, 2, 6]]
        },
        {
          "studentId": 2,
          "scoreRankIn3Year": [2, 3, [4, 3, 1]]
        }
      ]
    }
  } |
+-----+-----+
```

Extract JSON objects.

```
select json_tuple(school.json,"SchoolRank","Class1") as (item0,item1) from school;
-- The preceding and following statements are equivalent.
select get_json_object(school.json,"$.SchoolRank") item0,get_json_object(school.json,"$.Class1") item1 from school;
```

The following result is returned:

```
+-----+-----+
| item0 | item1 |
+-----+-----+
| 00    | {"Student":[{"studentId":1,"scoreRankIn3Year":[1,2,[3,2,6]]},{"studentId":2,"scoreRankIn3Year":[2,3,[4,3,1]]}] |
+-----+-----+
```

Parse JSON data that contains Chinese characters.

```
select json_tuple(school.json,"School name","Location") as (item0,item1) from school;
```

The following result is returned:

```
+-----+-----+
| item0 | item1 |
+-----+-----+
| Hupan college | Hangzhou |
+-----+-----+
```

Parse nested JSON data.

```
select sc.Id, q.item0, q.item1
from school sc LATERAL VIEW json_tuple(sc.json,"Class1.Student.[*].studentId","Class1.Student.[0].scoreRankIn3Year") q as item0,item1;
```

The following result is returned:

```
+-----+-----+-----+
| id   | item0 | item1 |
+-----+-----+-----+
| 1    | [1,2] | [1,2,[3,2,6]] |
+-----+-----+-----+
```

Parse JSON data that contains nested arrays.

```
select sc.Id, q.item0, q.item1
from school sc LATERAL VIEW json_tuple(sc.json,"Class1.Student[0].scoreRankIn3Year[2]","Class1.Student[0].scoreRankIn3Year[2][1]") q as item0,item1;
```

The following result is returned:

```
+-----+-----+-----+
| id   | item0 | item1 |
+-----+-----+-----+
| 1    | [3,2,6] | 2 |
+-----+-----+-----+
```

1.6.6.2.28.12. FROM_JSON

This topic describes the FROM_JSON function in string functions and provides examples.

Syntax:

```
FROM_JSON(jsonStr, schema)
```

Description: This function processes a JSON string and returns a value of the ARRAY, MAP, or STRUCT type with the specified schema.

Parameters:

- jsonStr: a value of the STRING type. This parameter specifies the JSON string to process.
- schema: the data structure. The syntax of schema is the same as that of table creation statements.

Examples:

- array<bigint>
- map<string, array<string>>
- struct<a:int, b:double, c:map<string,string>>

For the STRUCT type, you can also specify a value in the 'a INT, b DOUBLE' format, which is equivalent to struct<a:int, b:double> .

The following table describes the mapping relationships between JSON and MaxCompute data types.

JSON	MaxCompute
Object	STRUCT / MAP / STRING
Array	ARRAY / STRING
Number	TINYINT / SMALLINT / INT / BIGINT / FLOAT / DOUBLE / DECIMAL / STRING
true/false	BOOLEAN / STRING
String	STRING
null	All types

 **Note** For the OBJECT and ARRAY types, this function preferentially parses the data. For the types that are not supported by MaxCompute, this function ignores the data.

Return value: A value of the ARRAY, MAP, or STRUCT type is returned.

 **Note** If the schema parameter is left empty, an error is returned.

Examples:

```
SELECT from_json('{\"a\":1, \"b\":0.8}', 'a INT, b DOUBLE');
-- {\"a\":1, \"b\":0.8} is returned.
SELECT from_json('{\"time\":\"26/08/2015\"}', 'time string');
-- {\"time\":\"26/08/2015\"} is returned.
SELECT from_json('{\"a\":1, \"b\":0.8}', 'a INT, b DOUBLE, c STRING');
-- {\"a\":1, \"b\":0.8, c: NULL} is returned.
SELECT from_json('[1, 2, 3, \"a\"]', 'array<BIGINT>');
-- [1, 2, 3] is returned.
```

1.6.6.2.28.13. TO_JSON

This topic describes the TO_JSON function in string functions and provides examples.

Syntax:

```
TO_JSON(expr)
```

Description: This function returns a JSON string for a value of a given type.

Parameters: The expr parameter supports a value of the ARRAY, MAP, or STRUCT type.

Return value: A value of the STRING type is returned.

Examples:

```
SELECT to_json(named_struct('a', 1, 'b', 2));
-- {\"a\":1,\"b\":2} is returned.
SELECT to_json(named_struct('time', '26/08/2015'));
-- {\"time\":\"26/08/2015\"} is returned.
SELECT to_json(array(named_struct('a', 1, 'b', 2)));
-- [{\"a\":1,\"b\":2}] is returned.
SELECT to_json(map('a', named_struct('b', 1)));
-- {\"a\":{\"b\":1}} is returned.
SELECT to_json(map('a', 1));
-- {\"a\":1} is returned.
SELECT to_json(array((map('a', 1))));
-- [{\"a\":1}] is returned.
```

1.6.6.3. Date functions

1.6.6.3.1. DATEADD

This topic describes the date function DATEADD and provides examples.

Syntax:

```
datetime dateadd(datetime date, bigint delta, string datepart)
```

Description: This function modifies a date value based on datepart and delta that you specified.

Parameters:

- date: a date value of the DATE, DATETIME, or TIMESTAMP type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before the calculation. If the input value is of another data type, an error is returned.

- **delta**: a value of the BIGINT type, which indicates the interval to add to the specified part of the date value. If the input value is of the STRING or DOUBLE type, it is implicitly converted into a value of the BIGINT type before the calculation. If the input value is of another data type, an error is returned. If the value of delta is greater than 0, this function adds the interval to the date value. In other cases, this function subtracts the interval from the date value.

 Note

- If you add or subtract the interval specified by delta at a date part, a carry or return at more significant date parts may occur. The year, month, hour, minute, and second parts are computed by using different numeral systems. The year part uses the base-10 numeral system. The month part uses the base-12 numeral system. The hour part uses the base-24 numeral system. The minute and second parts use the base-60 numeral system.
- If the DATEADD function adds an interval specified by delta to the month part of a date value of the DATETIME type and this operation does not cause an overflow of day, keep day unchanged. If the operation causes an overflow of day, set day to the last day of the specified month.

- **datepart**: the part you want to modify in the date value. The value is a constant of the STRING type. If the value is in an invalid format or is not a constant of the STRING type, an error is returned. The value of this parameter is specified in compliance with the rules of conversions between the STRING and DATETIME types. The value yyyy indicates that the DATEADD function adds an interval to the year part of the date value. The value mm indicates that the DATEADD function adds an interval to the month part of the date value. This parameter also supports extended date formats, such as -year, -month, -mon, -day, and -hour. For more information about the rules of data type conversions, see [Conversion between string and datetime types](#).

Return value: A value of the DATETIME type is returned. If an input value is NULL, NULL is returned.

Examples:

- Example 1: common usage

```
select dateadd(datetime '2005-02-28 00:00:00', 1, 'dd');
-- The return value is 2005-03-01 00:00:00. After one day is added, the result is beyond the last day of February. The actual date value is the first day of March.
select dateadd(datetime '2005-02-28 00:00:00', -1, 'dd');
-- The return value is 2005-02-27 00:00:00. One day is subtracted.
select dateadd(datetime '2005-02-28 00:00:00', 20, 'mm');
-- The return value is 2006-10-28 00:00:00. After 20 months are added, the month overflows, and the year increases by 1.
select dateadd(datetime '2005-02-28 00:00:00', 1, 'mm');
-- The return value is 2005-03-28 00:00:00. One month is added.
select dateadd(datetime '2005-01-29 00:00:00', 1, 'mm');
-- The return value is 2005-02-28 00:00:00. February in 2005 has only 28 days. Therefore, the last day of February is returned.
select dateadd(datetime '2005-03-30 00:00:00', -1, 'mm');
-- The return value is 2005-02-28 00:00:00. One month is subtracted. February in 2005 has only 28 days. Therefore, the last day of February is returned.
```

- Example 2: usage of DATEADD in which a value of the DATETIME type is expressed as a constant

```
select dateadd(2005-03-30 00:00:00, -1, 'mm');
-- In MaxCompute SQL statements, a value of the DATETIME type cannot be directly expressed as a constant. This statement uses an invalid expression for a value of the DATETIME type.
```

```
select dateadd(cast("2005-03-30 00:00:00" as datetime), -1, 'mm');
-- This statement uses the DATEADD function in which a constant of the STRING type is explicitly converted to a value of the DATETIME type.
```

1.6.6.3.2. DATEDIFF

This topic describes the date function DATEDIFF and provides examples.

Syntax:

```
bigint datediff(datetime date1, datetime date2, string datepart)
```

Description: This function calculates the difference between date1 and date2. The difference is measured in the time unit specified by datepart.

Parameters:

- date1 and date2: the minuend and subtrahend. The values are of the DATE, DATETIME, or TIMESTAMP type. If an input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before the calculation. If the input value is of another data type, an error is returned.
- datepart: the time unit, which is a constant of the STRING type. This parameter supports extended date formats. If datepart is not in the specified format or is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If an input value is NULL, NULL is returned. If date1 is earlier than date2, a negative value is returned.

Note

- If the difference between two dates is more precise than the unit specified by datepart, the excessive parts are discarded in the return value. For example, if the unit specified by datepart is day, the hour, minute, and second parts are discarded in the return value.
- This function omits the parts with smaller units based on the unit specified by datepart and calculates the result. The parts with smaller units refer to the excessive parts previously described.

Examples:

```
-- The start time is 2005-12-31 23:59:59 and the end time is 2006-01-01 00:00:00.
datediff(end, start, 'dd') = 1
datediff(end, start, 'mm') = 1
datediff(end, start, 'yyyy') = 1
datediff(end, start, 'hh') = 1
datediff(end, start, 'mi') = 1
datediff(end, start, 'ss') = 1
datediff(datetime'2013-05-31 13:00:00', '2013-05-31 12:30:00', 'ss') = 1800
datediff(datetime'2013-05-31 13:00:00', '2013-05-31 12:30:00', 'mi') = 30
-- The start time is 2018-06-04 19:33:23.234 and the end time is 2018-06-04 19:33:23.250. Date values with milliseconds do not adopt the standard DATETIME type and therefore cannot be implicitly converted into the DATETIME type. In this case, an explicit conversion is required.
datediff(to_date('2018-06-04 19:33:23.250', 'yyyy-MM-dd hh:mi:ss.ff3'), to_date('2018-06-04 19:33:23.234', 'yyyy-MM-dd hh:mi:ss.ff3'), 'ff3') = 16
```

1.6.6.3.3. DATEPART

This topic describes the date function DATEPART and provides examples.

Syntax:

```
bigint datepart(datetime date, string datepart)
```

Description: This function returns a specific part of a date value. The part is specified by datepart.

Parameters:

- date: a date value of the DATE, DATETIME, or TIMESTAMP type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before the calculation. If the input value is of another data type, an error is returned.
- datepart: a constant of the STRING type. This parameter supports extended date formats. If datepart is not in the specified format or is of another data type, an error is returned.

Return value: A value of the BIGINT type is returned. If an input value is NULL, NULL is returned.

Examples:

```
datepart('2017-06-08 01:10:00', 'yyyy') = 2017  
datepart('2017-06-08 01:10:00', 'mm') = 6
```

1.6.6.3.4. DATETRUNC

This topic describes the date function DATETRUNC and provides examples.

Syntax:

```
datetime datetrunc (datetime date,string datepart)
```

Description: This function truncates a date value to the accuracy specified by datepart and returns a new date.

Parameters:

- date: a date value of the DATE, DATETIME, or TIMESTAMP type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before the calculation. If the input value is of another data type, an error is returned.
- datepart: a constant of the STRING type. This parameter supports extended date formats. If datepart is not in the specified format or is of another data type, an error is returned.

Return value: A value of the DATETIME type is returned. If an input value is NULL, NULL is returned.

Examples:

```
datetrunc('2017-12-07 16:28:46', 'yyyy') = 2017-01-01 00:00:00  
-- The function truncates the date value to the accuracy of the year 2017 and returns the date value that is accurate to the year. In this example, the date value that is accurate to the year 2017 is returned.  
datetrunc('2017-12-07 16:28:46', 'month') = 2017-12-01 00:00:00  
-- The function truncates the date value to the accuracy of the month December and returns the date value that is accurate to the month. In this example, the date value that is accurate to the month "2017-12" is returned.  
datetrunc('2017-12-07 16:28:46', 'DD') = 2017-12-07 00:00:00  
-- The function truncates the date value to the accuracy of the day 07 and returns the date value that is accurate to the day. In this example, the date value that is accurate to the day "2017-12-07" is returned.
```

1.6.6.3.5. GETDATE

This topic describes the GETDATE function in date functions.

Syntax:

```
datetime getdate()
```

Description: This function returns the current system time as a date value. MaxCompute uses UTC+8 as the standard time zone.

Return value: A value of the DATETIME type is returned. The value indicates the current date and time.

 **Note** In a MaxCompute SQL task that is executed in distributed mode, the GETDATE function always returns a fixed value. The return value is an arbitrary time during the execution of the MaxCompute SQL task. The time is accurate to seconds. In MaxCompute V2.0 that supports more data types, the time is accurate to milliseconds.

1.6.6.3.6. ISDATE

This topic describes the ISDATE function in date functions.

Syntax:

```
boolean isdate(string date, string format)
```

Description: This function determines whether a date string can be converted into a date value in a specified format. If the date string can be converted into a date value in the specified format, True is returned. Otherwise, False is returned.

Parameters:

- **date**: a value of the STRING type. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type, an error is returned.
- **format**: a constant of the STRING type. This parameter does not support extended date formats. If the input value is of another data type, an error is returned. If redundant format strings exist in format, this function converts the date string that corresponds to the first format string into a date value. The rest strings are considered delimiters. For example, `isdate("1234-yyyy", "yyyy-yyyy")` returns True.

Return value: A value of the BOOLEAN type is returned. If an input value is NULL, NULL is returned.

1.6.6.3.7. LASTDAY

This topic describes the LASTDAY function in date functions.

Syntax:

```
datetime lastday(datetime date)
```

Description: This function returns the last day of the month in which the date value falls. The value is accurate to days. The hour, minute, and second parts are expressed as 00:00:00.

Parameters: The date parameter supports a value of the DATETIME type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before calculation. If the input value is of another data type, an error is returned.

Return value: A value of the DATETIME type is returned. If the input value is NULL, NULL is returned.

1.6.6.3.8. TO_DATE

This topic describes the TO_DATE function and provides examples of using this function.

Syntax:

```
datetime to_date(string date, string format)
```

Description: This function converts a date string in a specified format into a date value.

Parameters:

- **date**: a date value of the STRING type, which indicates the date string you want to convert. If the input value is of the BIGINT, DOUBLE, DECIMAL, or DATETIME type, it is implicitly converted into a value of the STRING type before calculation. If the input value is of another data type or an empty string, an error is returned.
- **format**: a constant of the STRING type, which indicates a date format. If the input value is not a constant or is not of the STRING type, an error is returned. The Extended Date/Time Format (EDTF) is not supported. In this format, the characters are parsed as invalid characters and omitted. The value of this parameter must contain yyyy. Otherwise, an error is returned. If redundant format strings are included in the value of this parameter, only the date value that corresponds to the first format string is used, and the date values that correspond to the other format strings are parsed as delimiters. For example, to_date('1234-2234 ', 'yyyy-yyyy ') is parsed as 1234-01-01 00:00:00.

Return value: A value of the DATETIME type is returned and the value is in the format of yyyy-mm-dd hh:mi:ss:ff3. If the value of any input parameter is NULL, NULL is returned.

 **Note** In the format, yyyy indicates a 4-digit year, mm indicates a 2-digit month, dd indicates a 2-digit day, hh indicates an hour in 24-hour display, mi indicates a 2-digit minute, ss indicates a 2-digit second, and ff3 indicates a 3-digit millisecond.

Examples:

```
to_date('Alibaba2017-12*03', 'Alibayyyy-mm*dd') = 2017-12-03 00:00:00
to_date('20170718', 'yyyymmdd') = 2017-07-18 00:00:00
to_date('201707182030', 'yyyymmddhhmi') = 2017-07-18 20:30:00
to_date('2017718', 'yyyymmdd')
-- Invalid format. NULL is returned.
to_date('Alibaba2017-12*3', 'Alibayyyy-mm*dd')
-- Invalid format. NULL is returned.
to_date('2017-24-01', 'yyyy')
-- Invalid format. NULL is returned.
to_date('20181030 15-13-12.345', 'yyyymmdd hh-mi-ss.ff3') = 2018-10-30 15:13:12
```

1.6.6.3.9. TO_CHAR

This topic describes the TO_CHAR function and provides examples of using this function.

Syntax:

```
string to_char(datetime date, string format)
```

Description: This function converts a date value of the DATETIME type into a string in a specified format.

Parameters:

- **date**: a date value of the DATETIME type, which indicates the date value you want to convert. If the input value is of the STRING type, it is implicitly converted into the DATETIME type before calculation. If the input value is not of the STRING type, an error is returned.

- **format**: a constant of the STRING type. If it is not a constant or is not of the STRING type, an error is returned. In the format parameter, the date format part is replaced by the related data and other characters remain unchanged in the output.

Return value: A value of the STRING type is returned. If the value of an input parameter is NULL, NULL is returned.

Examples:

```
to_char('2017-12-03 00:00:00', 'Alibaba Cloud Financial Servicesyyyy-mm*dd') = 'Alibaba Cloud Financial Services2017-12*03'
to_char('2017-07-18 00:00:00', 'yyyymmdd') = '20170718'
to_char('Alibaba2017-12*3', 'Alibabayyy-mm*dd')
-- NULL is returned.
to_char('2017-24-01', 'yyyy')
-- NULL is returned.
to_char('2017718', 'yyyymmdd')
-- NULL is returned.
```

1.6.6.3.10. UNIX_TIMESTAMP

This topic describes the UNIX_TIMESTAMP function and provides examples of using this function.

Syntax:

```
bigint unix_timestamp(datetime date)
```

Description: This function converts a date value to a UNIX timestamp that is an integer.

Parameters: **date**: a date value of the DATETIME type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before calculation. If the input value is not of the STRING type, an error is returned. If you enable new data types, the implicit conversion fails. In this case, you must use the CAST function for conversion, for example, `unix_timestamp(cast(... as datetime))`.

Return value: A UNIX timestamp of the BIGINT type is returned. If the value of the input parameter is NULL, NULL is returned.

Examples:

```
select unix_timestamp(datetime'2009-03-20 11:11:00');
-- 1237518660 is returned.
```

1.6.6.3.11. FROM_UNIXTIME

This topic describes the FROM_UNIXTIME function and provides examples of using this function.

Syntax:

```
datetime from_unixtime(bigint unixtime)
```

Description: This function converts unixtime of the BIGINT type to a date value of the DATETIME type.

Parameters: **unixtime**: a date value of the BIGINT type in the UNIX format. Its value is accurate to seconds. If the input value is of the STRING, DOUBLE, or DECIMAL type, it is implicitly converted into a value of the BIGINT type before calculation.

Return value: A value of the DATETIME type is returned. If an input value is NULL, NULL is returned.

 **Note** In the Hive-compatible mode where `set odps.sql.hive.compatible=true;` has been run, if the value of the input parameter is of the STRING type, a date value of the STRING type is returned.

Examples:

```
from_unixtime(123456789) = 1973-11-30 05:33:09;
```

1.6.6.3.12. WEEKDAY

This topic describes the WEEKDAY function.

Syntax:

```
bigint weekday (datetime date)
```

Description: This function returns the day of the week for a specified date.

Parameters: date: a date value of the DATETIME type. If the input value is of the STRING type, it is implicitly converted into a value of the DATETIME type before calculation. If the input value is not of the STRING type, an error is returned.

Return value: A value of the BIGINT type is returned. If the value of the input parameter is NULL, NULL is returned. Monday is treated as the first day of a week and its return value is 0. Days are numbered in ascending order starting from 0. The return value for Sunday is 6.

1.6.6.3.13. WEEKOFYEAR

This topic describes the WEEKOFYEAR function and provides examples of using this function.

Syntax:

```
bigint weekofyear(datetime date)
```

Description: This function returns the calendar week of the year that the specified date falls in. Monday is treated as the first day of a week.

 **Note** If a week spans two years, whether this week belongs to the previous year or the next year is based on which year contains more than four days. If more days fall in the previous year, the week is considered as the last week of the previous year. If more days fall in the next year, the week is considered as the first week of the next year.

Parameters: date: a date value of the DATETIME type. If the value of the input parameter is of the STRING type, it is implicitly converted into a value of the DATETIME type before calculation. If the value of the input parameter is not of the STRING type, an error is returned.

Return value: A value of the BIGINT type is returned. If the value of the input parameter is NULL, NULL is returned.

Examples:

```
select weekofyear(to_date("20141229", "yyyymmdd"));
-- The following result is returned:
+-----+
|_c0 |
+-----+
| 1 |
+-----+
-- 20141229 is in year 2014, but most days of the week fall in year 2015. Therefore, the return value 1 indicates the first week of year 2015.
select weekofyear(to_date("20141231", "yyyymmdd"));
-- 1 is returned.
select weekofyear(to_date("20151229", "yyyymmdd"));
-- 53 is returned.
```

1.6.6.3.14. Additional date functions

1.6.6.3.14.1. Usage notes

This topic describes the configuration operation that you must perform before you use the date functions that are added to MaxCompute V2.0.

MaxCompute V2.0 provides more date functions. You must add the following SET statement before the SQL statements for these new functions:

```
set odps.sql.type.system.odps2=true;
```

 **Note** You must submit and execute the SET statement and the SQL statements for the new functions at the same time.

1.6.6.3.14.2. YEAR

This topic describes the YEAR function and provides examples of using this function.

Syntax:

```
INT YEAR(DATETIME/STRING date)
```

Description: This function returns the year in which the specified date value falls.

Parameters: date: a date value of the DATETIME or STRING type. The format of the date value must include yyyy-mm-dd and exclude redundant strings. Otherwise, NULL is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT YEAR('1970-01-01 12:30:00');
-- 1970 is returned.
SELECT YEAR('1970-01-01');
-- 1970 is returned.
SELECT YEAR('70-01-01');
-- 70 is returned.
SELECT YEAR('1970-01-01');
-- 1970 is returned.
SELECT YEAR('1970/03/09');
-- NULL is returned.
SELECT YEAR(null);
-- An error is returned.
```

1.6.6.3.14.3. QUARTER

This topic describes the QUARTER function and provides examples of using this function.

Syntax:

```
INT QUARTER (DATETIME/TIMESTAMP/STRING date)
```

Description: This function returns the quarter of the year for a date value. The quarter is an integer from 1 to 4.

Parameters: date: a date value of the DATETIME, TIMESTAMP, or STRING type. The format of the date value must include yyyy-mm-dd. If the date value is not of the DATETIME, TIMESTAMP, or STRING type, NULL is returned.

Return value: A value of the INT type is returned. If the value of an input parameter is NULL, NULL is returned.

Examples:

```
SELECT QUARTER('1970-11-12 10:00:00');
-- 4 is returned.
SELECT QUARTER('1970-11-12');
-- 4 is returned.
```

1.6.6.3.14.4. MONTH

This topic describes the MONTH function and provides examples of using this function.

Syntax:

```
INT MONTH(DATETIME/STRING date)
```

Description: This function returns the month part of a date value.

Parameters: date: a value of the DATETIME or STRING type. The date format must include yyyy-mm-dd. If it is not of the DATETIME or STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT MONTH('2014-09-01');  
-- 9 is returned.  
SELECT MONTH('20140901');  
-- NULL is returned.
```

1.6.6.3.14.5. DAY

This topic describes the DAY function and provides examples of using this function.

Syntax:

```
INT DAY(DATETIME/STRING date)
```

Description: This function returns the day of a specified date value.

Parameters: date: a date value of the DATETIME or STRING type. The format of the date value can be yyyy-mm-dd or yyyy-mm-dd hh:mi:ss. If the date value is not of the DATETIME or STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT DAY('2014-09-01');  
-- 1 is returned.  
SELECT DAY('20140901');  
-- NULL is returned.
```

1.6.6.3.14.6. DAYOFMONTH

This topic describes the DAYOFMONTH function and provides examples of using this function.

Syntax:

```
int dayofmonth(date)
```

Description: This function returns the value of the day part of a date value. For example, if the date is October 13, 2019, 13 is returned after you run the `int dayofmonth(2019-10-13)` command.

Parameters: date: a date value of the STRING type. The date format must include yyyy-mm-dd. If it is not of the STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
dayofmonth('2019-09-01');  
-- 1 is returned.  
dayofmonth('20190901');  
-- NULL is returned.
```

1.6.6.3.14.7. HOUR

This topic describes the HOUR function and provides examples of using this function.

Syntax:

INT HOUR(DATETIME/STRING date)

Description: This function returns the value of the hour part of a date.

Parameters: date: a value of the DATETIME or STRING type. If the value is not of the DATETIME or STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT HOUR('2014-09-01 12:00:00');  
-- 12 is returned.  
SELECT HOUR('12:00:00');  
-- 12 is returned.  
SELECT HOUR('20140901120000');  
-- NULL is returned.
```

1.6.6.3.14.8. MINUTE

This topic describes the MINUTE function and provides examples of using this function.

Syntax:

INT MINUTE(DATETIME/STRING date)

Description: This function returns the value of the minute part of a date.

Parameters: date: a value of the DATETIME or STRING type. If the value is not of the DATETIME or STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT MINUTE('2014-09-01 12:30:00');  
-- 30 is returned.  
SELECT MINUTE('12:30:00');  
-- 30 is returned.  
SELECT MINUTE('20140901120000');  
-- NULL is returned.
```

1.6.6.3.14.9. SECOND

This topic describes the SECOND function and provides examples of using this function.

Syntax:

INT SECOND(DATETIME/STRING date)

Description: This function returns the value of the second part of a date value.

Parameters: date: a date value of the DATETIME or STRING type. If it is not of the DATETIME or STRING type, an error is returned.

Return value: A value of the INT type is returned.

Examples:

```
SELECT SECOND('2014-09-01 12:30:45');  
-- 45 is returned.  
SELECT SECOND('12:30:45');  
-- 45 is returned.  
SELECT SECOND('20140901123045');  
-- NULL is returned.
```

1.6.6.3.14.10. FROM_UTC_TIMESTAMP

This topic describes the FROM_UTC_TIMESTAMP function and provides examples of using this function.

Syntax:

```
timestamp from_utc_timestamp({any primitive type}*, string timezone)
```

Description: This function converts a UTC timestamp to a timestamp for a specified time zone.

Parameters:

- {any primitive type}*: the timestamp, which is a value of the TIMESTAMP, DATETIME, TINYINT, SMALLINT, INT, or BIGINT type. If the value is of the TINYINT, SMALLINT, INT, or BIGINT type, the unit is milliseconds.
- timezone: the time zone to which you want to convert the timestamp, such as Pacific Standard Time (PST).

Return value: A value of the TIMESTAMP type is returned.

Examples:

```
SELECT from_utc_timestamp(1501557840000, 'PST');  
-- The unit of the input value is milliseconds and 2017-08-01 04:24:00 is returned.  
SELECT from_utc_timestamp('1970-01-30 16:00:00', 'PST');  
-- 1970-01-30 08:00:00.0 is returned.  
SELECT from_utc_timestamp('1970-01-30', 'PST');  
-- 1970-01-29 16:00:00.0 is returned.
```

1.6.6.3.14.11. CURRENT_TIMESTAMP

This topic describes the CURRENT_TIMESTAMP function and provides examples of using this function.

Syntax:

```
timestamp current_timestamp()
```

Description: This function returns the current timestamp. The return value is not fixed.

Return value: A value of the TIMESTAMP type is returned.

Examples:

```
select current_timestamp();  
-- '2017-08-03 11:50:30.661' is returned.
```

1.6.6.3.14.12. ADD_MONTHS

This topic describes the ADD_MONTHS function and provides examples of using this function.

Syntax:

```
string add_months(string startdate, int nummonths)
```

Description: This function adds a specified number of months to a date specified by startdate. The number of months is specified by nummonths.

Parameters:

- startdate: a date value of the STRING type. The date format must contain yyyy-mm-dd. Otherwise, NULL is returned.
- num_months: a value of the INT type.

Return value: A value of the STRING type is returned and the value is in the format of yyyy-mm-dd.

Examples:

```
add_months('2017-02-14',3) = '2017-05-14'  
add_months('17-2-14',3) = '0017-05-14'  
add_months('2017-02-14 21:30:00',3) = '2017-05-14'  
add_months('20170214',3) = null
```

1.6.6.3.14.13. LAST_DAY

This topic describes the LAST_DAY function and provides examples of using this function.

Syntax:

```
string last_day(string date)
```

Description: This function returns the last date of the month part of a date value.

Parameters: date: a date value of the STRING type and in the format of yyyy-mm-dd hh:mi:ss or yyyy-mm-dd.

Return value: A value of the STRING type is returned and the value is in the format of yyyy-mm-dd.

Examples:

```
last_day('2017-03-04') = '2017-03-31'  
last_day('2017-07-04 11:40:00') = '2017-07-31'  
last_day('20170304') = null
```

1.6.6.3.14.14. NEXT_DAY

This topic describes the NEXT_DAY function and provides examples of using this function.

Syntax:

```
string next_day(string startdate, string week)
```

Description: This function returns the date of the first day that is later than startdate and matches the week value. The return value indicates the date of the specified day in the next week.

Parameters:

- startdate: a date value of the STRING type and in the format of yyyy-mm-dd hh:mi:ss or yyyy-mm-dd.
- week: a date value of the STRING type. The full name of a day in a week, or the first two or three letters of

the day, for example, MO, TUE, or FRIDAY.

Return value: A value of the STRING type is returned and the value is in the format of yyyy-mm-dd.

Examples:

```
next_day('2017-08-01','TU') = '2017-08-08'
next_day('2017-08-01 23:34:00','TU') = '2017-08-08'
next_day('20170801','TU') = null
```

1.6.6.3.14.15. MONTHS_BETWEEN

This topic describes the MONTHS_BETWEEN function and provides examples of using this function.

Syntax:

```
double months_between(datetime/timestamp/string date1, datetime/timestamp/string date2)
```

Description: This function returns the number of months between date1 and date2.

Parameters:

- date1: a value of the DATETIME, TIMESTAMP, or STRING type. The value is in the format of yyyy-mm-dd hh:mi:ss or yyyy-mm-dd.
- date2: a value of the DATETIME, TIMESTAMP, or STRING type. The value is in the format of yyyy-mm-dd hh:mi:ss or yyyy-mm-dd.

Return value: A value of the DOUBLE type is returned.

- If date1 is later than date2, a positive value is returned. If date2 is later than date1, a negative value is returned.
- If both date1 and date2 correspond to the last days of two months, the return value is an integer that represents the number of months. Otherwise, the return value is computed by using the following formula: $(\text{date1} - \text{date2}) / 31$

Examples:

```
months_between('1997-02-28 10:30:00', '1996-10-30') = 3.9495967741935485
months_between('1996-10-30', '1997-02-28 10:30:00') = -3.9495967741935485
months_between('1996-09-30', '1996-12-31') = -3.0
```

1.6.6.3.14.16. EXTRACT

This topic describes the EXTRACT function and provides examples of using this function.

Syntax:

```
int extract(<datepart> from <timestamp>)
```

Description: This function extracts the part specified by datepart from the time specified by timestamp.

Parameters:

- datepart: a value that can be set to YEAR, MONTH, DAY, HOUR, or MINUTE.
- timestamp: a date value of the TIMESTAMP type.

Return value: A value of the INT type is returned.

Examples:

```
set odps.sql.type.system.odps2=true;
select extract(year from '2019-05-01 11:21:00') year
       ,extract(month from '2019-05-01 11:21:00') month
       ,extract(day from '2019-05-01 11:21:00') day
       ,extract(hour from '2019-05-01 11:21:00') hour
       ,extract(minute from '2019-05-01 11:21:00') minute;
-- The following result is returned:
+-----+-----+-----+-----+-----+
| year | month | day | hour | minute |
+-----+-----+-----+-----+-----+
| 2019 | 5    | 1  | 11  | 21    |
+-----+-----+-----+-----+-----+
```

If the time value specified in the SQL statement is invalid or exceeds the specified range, the return value is the remainder obtained by dividing the specified time value by the maximum value in the time range.

Examples:

```
set odps.sql.type.system.odps2=true;
select extract(hour from '2019-05-01 31:01:01') hour
       ,extract(minute from '2019-05-01 23:61:01') minute;
-- The following result is returned:
+-----+-----+
| hour | minute |
+-----+-----+
| 7    | 1      |
+-----+-----+
-- The maximum value of the hour part is 24, and the specified time value is 31. The return value is 7 (31/24).
-- The maximum value of the minute part is 60, and the specified time value is 61. The return value is 1 (61/60).
```

1.6.6.4. Window functions

1.6.6.4.1. Overview

In MaxCompute SQL, window functions can be used for flexible data analysis and processing. This topic describes the precautions for using window functions and the syntax of window functions.

Notice

- Window functions can only be included in the SELECT clause.
- A window function cannot contain nested window functions or aggregate functions.
- Window functions cannot be used with aggregate functions of the same level.
- A maximum of five window functions can be used in a MaxCompute SQL statement.

Syntax:

```
window_func() over (partition by [col1,col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] windowing_clause)
```

Parameters:

- PARTITION BY: the columns in the window that is used for calculation. Rows with the same values of the

partition key columns are considered in the same window. A window can contain a maximum of 100 million rows of data. We recommend that the number of rows in a window not exceed 5 million. If the number of rows exceeds 5 million, an error is returned.

- ORDER BY: specifies how to sort data in a window.
- windowing_clause: You can use ROWS to specify how to define the window for the function. Valid values:
 - rows between x preceding|following and y preceding|following: indicates a window that ranges from the xth row preceding or following the current row to the yth row preceding or following the current row.
 - rows x preceding|following: indicates a window that ranges from the xth row preceding or following the current row to the current row.

Note

- x and y must be integer constants greater than or equal to 0. Their values range from 0 to 10000. 0 indicates the current row.
- You must specify ORDER BY before you use ROWS to specify a window range.
- Window functions that allow you to use rows to define the window include AVG, COUNT, MAX, MIN, STDDEV, and SUM.

1.6.6.4.2. COUNT

This topic describes the COUNT function and provides examples of using this function.

Syntax:

```
Bigint count([distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function returns the counted values of specified rows.

Parameters:

- expr: a value of any data type. If the value for a row is NULL, this row is not used for calculation. If the distinct keyword is specified, only the distinct values are counted.
- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: If ORDER BY is not specified, the counted value of expr in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the value accumulated from the starting row to the current row in the current window is returned.

 Note If the distinct keyword is specified, ORDER BY cannot be used.

Return value: A value of the BIGINT type is returned.

Examples:

The test_src table contains the user_id column of the BIGINT type.

```
select user_id,count(user_id) over (partition by user_id) as count from test_src;
```

```
+-----+-----+
|user_id|count |
+-----+-----+
| 1 | 3 |
| 1 | 3 |
| 1 | 3 |
| 2 | 1 |
| 3 | 1 |
+-----+-----+
```

-- If ORDER BY is not specified, the counted value of the user_id column in the current window is returned.

```
select user_id,count(user_id) over (partition by user_id order by user_id) as count from test_src;
```

```
+-----+-----+
|user_id|count |
+-----+-----+
| 1 | 1 |
| 1 | 2 |
| 1 | 3 |
| 2 | 1 |
| 3 | 1 |
+-----+-----+
```

-- This row is the starting row of this window.

-- Two rows are found from the starting row to the current row. 2 is returned.

-- If ORDER BY is specified, the counted value from the starting row to the current row in the current window is returned.

If duplicate values are specified for ORDER BY, the processing method is based on the compatibility between MaxCompute and Hive.

- If MaxCompute is not compatible with Hive, the value of COUNT for each row is returned.

```
set odps.sql.hive.compatible=false;
select user_id, price, count(price) over
(partition by user_id order by price) as count from test_src;
```

```
+-----+-----+-----+
|user_id|price |count |
+-----+-----+-----+
| 1 | 4.5 | 1 | -- This row is the starting row of this window.
| 1 | 5.5 | 2 | -- The value of COUNT for the second row is 2.
| 1 | 5.5 | 3 | -- The value of COUNT for the third row is 3.
| 1 | 6.5 | 4 |
| 2 | NULL | 0 |
| 2 | 3.0 | 1 |
| 3 | NULL | 0 |
| 3 | 4.0 | 1 |
+-----+-----+-----+
```

- If MaxCompute is compatible with Hive, the return value is the value of COUNT for the last row of the rows with the same value.

```

set odps.sql.hive.compatible=true;
select user_id, price, count(price) over
(partition by user_id order by price) as count from test_src;
+-----+-----+-----+
| user_id | price | count |
+-----+-----+-----+
| 1 | 4.5 | 1 | -- This row is the starting row of this window.
| 1 | 5.5 | 3 | -- The value of COUNT for the second row is the same as the value of COUNT for the third row because the prices of the two rows are the same.
| 1 | 5.5 | 3 | -- The value of COUNT for the third row is 3.
| 1 | 6.5 | 4 |
| 2 | NULL | 0 |
| 2 | 3.0 | 1 |
| 3 | NULL | 0 |
| 3 | 4.0 | 1 |
+-----+-----+-----+

```

1.6.6.4.3. AVG

This topic describes the AVG function and provides examples of using this function.

Syntax:

```

avg([distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])

```

Description: This function returns the average value of input values in specified rows.

Parameters:

- **expr**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned. If the input value for a row is NULL, the row is not used for calculation. Values of the BOOLEAN type are not used for calculation. If the distinct keyword is specified, the average value of distinct values is calculated.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.
- **order by [col1[asc|desc], col2[asc|desc]**: If ORDER BY is not specified, the average value of all values in the current window is returned. If ORDER BY is specified, the returned results are sorted in a specified order and the accumulated average value of the values from the starting row to the current row is returned.

 **Note** If the distinct keyword is specified, ORDER BY cannot be used.

Return value: A value of the DOUBLE type is returned.

If duplicate values are specified for ORDER BY, the processing method is based on the compatibility between MaxCompute and Hive.

- If MaxCompute is not compatible with Hive, the return value is the average value for each row.

```

set odps.sql.hive.compatible=false;
select user_id, price, avg(price) over
(partition by user_id order by price) from test_src;
+-----+-----+-----+
| user_id | price | _c2      |
+-----+-----+-----+
| 1       | 4.5   | 4.5      | -- This row is the starting row of this window.
| 1       | 5.5   | 5.0      | -- The return value is the average value of the values for the first and second rows.
| 1       | 5.5   | 5.166666666666667 | -- The return value is the average value of the values from the first row to t
he third row.
| 1       | 6.5   | 5.5      |
| 2       | NULL  | NULL     |
| 2       | 3.0   | 3.0      |
| 3       | NULL  | NULL     |
| 3       | 4.0   | 4.0      |
+-----+-----+-----+
    
```

- If MaxCompute is compatible with Hive, the return value is the average value of values for the last row of the rows with the same value.

```

set odps.sql.hive.compatible=true;
select user_id, price, avg(price) over
(partition by user_id order by price) from test_src;
+-----+-----+-----+
| user_id | price | _c2      |
+-----+-----+-----+
| 1       | 4.5   | 4.5      | -- This row is the starting row of this window.
| 1       | 5.5   | 5.166666666666667 | -- The return value is the average value of values from the first row to the thi
rd row.
| 1       | 5.5   | 5.166666666666667 | -- The return value is the average value of values from the first row to the thi
rd row.
| 1       | 6.5   | 5.5      |
| 2       | NULL  | NULL     |
| 2       | 3.0   | 3.0      |
| 3       | NULL  | NULL     |
| 3       | 4.0   | 4.0      |
+-----+-----+-----+
    
```

1.6.6.4.4. MAX

This topic describes the MAX function.

Syntax:

```

max([distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
    
```

Description: This function calculates the maximum value of specified rows in the current window.

Parameters:

- expr: a value of any data type except BOOLEAN. If the value for a row is NULL, this row is not used for calculation. If the distinct keyword is specified, the maximum value of distinct values is calculated. The calculation result is not affected regardless of whether the parameter is specified.
- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: If ORDER BY is not specified, the maximum value of all values in the

current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the maximum value of values from the starting row to the current row in the current window is returned.

 **Note** If the distinct keyword is specified, ORDER BY cannot be used.

Return value: A value of the same data type as expr is returned.

1.6.6.4.5. MIN

This topic describes the MIN function.

Syntax:

```
min([distinct] expr) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function returns the minimum value of values in specified rows.

Parameters:

- **expr**: a value of any data type except BOOLEAN. If the value for a row is NULL, the row is not used for calculation. If the distinct keyword is specified, the minimum value of distinct values is calculated. The calculation result is not affected regardless of whether the parameter is specified.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.
- **order by [col1[asc|desc], col2[asc|desc]**: If ORDER BY is not specified, the minimum value in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the minimum value of values from the starting row to the current row in the current window is returned.

 **Note** If the distinct keyword is specified, ORDER BY cannot be used.

Return value: A value of the same data type as expr is returned.

1.6.6.4.6. MEDIAN

This topic describes the MEDIAN function.

Syntax:

```
double median(double number1,number2...) over(partition by [col1, col2...])  
decimal median(decimal number1,number2...) over(partition by [col1, col2...])
```

Description: This function calculates the median.

Parameters:

- **number1,number2...**: numbers of the DOUBLE or DECIMAL type. You can enter 1 to 255 numbers. At least one number is required. If the input value is of the DOUBLE type, it is automatically converted into an array of the DOUBLE type before calculation. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If it is not of the STRING or BIGINT type, an error is returned. If the input value is NULL, NULL is returned.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.

Return value: A value of the DOUBLE or DECIMAL type is returned.

1.6.6.4.7. STDDEV

This topic describes the STDDEV function and provides examples of using this function.

Syntax:

```
Double stddev([distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
Decimal stddev([distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function calculates the population standard deviation of values in specified rows.

Parameters:

- **expr**: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned. If the value for a row is NULL, this row is not used for calculation. If the distinct keyword is specified, the population standard deviation of distinct values is calculated.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.
- **order by [col1[asc|desc], col2[asc|desc]**: If ORDER BY is not specified, the population standard deviation of rows in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the population standard deviation from the starting row to the current row in the current window is returned.

 **Note** If the distinct keyword is specified, ORDER BY cannot be used.

Return value: If the input value is of the DECIMAL type, a value of the DECIMAL type is returned. Otherwise, a value of the DOUBLE type is returned.

Examples:

```
select window, seq, stddev_pop('1\01') over (partition by window order by seq) from dual;
```

If duplicate values are specified for ORDER BY, the processing method is based on the compatibility between MaxCompute and Hive.

- If MaxCompute is not compatible with Hive, the return value is the value of STDDEV for each row.

```
set odps.sql.hive.compatible=false;
select user_id, price, stddev(price) over
(partition by user_id order by price) from test_src;
+-----+-----+-----+
| user_id | price | _c2 |
+-----+-----+-----+
| 1 | 4.5 | 0.0 | -- This row is the starting row of this window.
| 1 | 5.5 | 0.5 | -- The return value is the value of STDDEV for the first and second rows.
| 1 | 5.5 | 0.4714045207910316 | -- The return value is the value of STDDEV from the first row to the third row.
| 1 | 6.5 | 0.7071067811865475 |
| 2 | NULL | NULL |
| 2 | 3.0 | 0.0 |
| 3 | NULL | NULL |
| 3 | 4.0 | 0.0 |
+-----+-----+-----+
```

- If MaxCompute is compatible with Hive, the return value is the value of STDDEV for the last row of the rows with the same value.

```
set odps.sql.hive.compatible=true;
select user_id, price, stddev(price) over
(partition by user_id order by price) from test_src;
+-----+-----+-----+
| user_id | price | _c2      |
+-----+-----+-----+
| 1       | 4.5   | 0.0      | -- This row is the starting row of this window.
| 1       | 5.5   | 0.4714045207910316 | -- The return value is the value of STDDEV from the first row to the third row.
| 1       | 5.5   | 0.4714045207910316 | -- The return value is the value of STDDEV from the first row to the third row.
| 1       | 6.5   | 0.7071067811865475 |
| 2       | NULL  | NULL     |
| 2       | 3.0   | 0.0      |
| 3       | NULL  | NULL     |
| 3       | 4.0   | 0.0      |
+-----+-----+-----+
```

1.6.6.4.8. STDDEV_SAMP

This topic describes the STDDEV_SAMP function.

Syntax:

```
Double stddev_samp([(distinct] expr) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
Decimal stddev_samp([(distinct] expr) over((partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function calculates the sample standard deviation.

Parameters:

- `expr`: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned. If the value for a row is NULL, the row is not used for calculation. If the `distinct` keyword is specified, the sample standard deviation of distinct values is calculated.
- `partition by [col1, col2...]`: the columns in the window that is used for calculation.
- `order by [col1[asc|desc], col2[asc|desc]`: If ORDER BY is not specified, the sample standard deviation in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the sample standard deviation from the starting row to the current row in the current window is returned.

 **Note** If the `distinct` keyword is specified, ORDER BY cannot be used.

Return value: If the input value is of the DECIMAL type, a value of the DECIMAL type is returned. Otherwise, a value of the DOUBLE type is returned.

1.6.6.4.9. SUM

This topic describes the SUM function and provides examples of using this function.

Syntax:

```
sum([distinct] expr) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function returns the sum of values in specified rows.

Parameters:

- **expr**: a value of the DOUBLE, DECIMAL, or BIGINT type. If the input value is of the STRING type, it is implicitly converted to the DOUBLE type before calculation. If it is not of the STRING type, an error is returned. If the value for a row is NULL, the row is not used for calculation. If the distinct keyword is specified, the sum of distinct values is calculated.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.
- **order by [col1[asc|desc], col2[asc|desc]**: If ORDER BY is not specified, the sum of the expr values in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the sum of the values from the starting row to the current row in the current window is returned.

 **Note** If the distinct keyword is specified, ORDER BY cannot be used.

Return value: If the input value is of the BIGINT type, a value of the BIGINT type is returned. If the input value is of the DECIMAL type, a value of the DECIMAL type is returned. If the input value is of the DOUBLE or STRING type, a value of the DOUBLE type is returned.

If duplicate values are specified for ORDER BY, the processing method is based on the compatibility between MaxCompute and Hive.

- If MaxCompute is not compatible with Hive, the return value is the sum of values for each row.

```
set odps.sql.hive.compatible=false;  
select user_id, price, sum(price) over  
  (partition by user_id order by price) from test_src;
```

```
+-----+-----+-----+  
|user_id|price|_c2|  
+-----+-----+-----+
```

```
| 1 | 4.5 | 4.5 | -- This row is the starting row of this window.  
| 1 | 5.5 | 10.0 | -- The return value is the sum of values for the first and second rows.  
| 1 | 5.5 | 15.5 | -- The return value is the sum of values from the first row to the third row.  
| 1 | 6.5 | 22.0 |  
| 2 | NULL | NULL |  
| 2 | 3.0 | 3.0 |  
| 3 | NULL | NULL |  
| 3 | 4.0 | 4.0 |  
+-----+-----+-----+
```

- If MaxCompute is compatible with Hive, the return value is the sum of values for the last row of the rows with the same value.

```
set odps.sql.hive.compatible=true;
select user_id, price, sum(price) over
(partition by user_id order by price) from test_src;
```

```
+-----+-----+-----+
| user_id | price | _c2 |
+-----+-----+-----+
| 1 | 4.5 | 4.5 | -- This row is the starting row of this window.
| 1 | 5.5 | 15.5 | -- The return value is the sum of values from the first row to the third row.
| 1 | 5.5 | 15.5 | -- The return value is the sum of values from the first row to the third row.
| 1 | 6.5 | 22.0 |
| 2 | NULL | NULL |
| 2 | 3.0 | 3.0 |
| 3 | NULL | NULL |
| 3 | 4.0 | 4.0 |
+-----+-----+-----+
```

1.6.6.4.10. DENSE_RANK

This topic describes the DENSE_RANK function and provides examples of using this function.

Syntax:

```
Bigint dense_rank() over(partition by [col1, col2...]
order by [col1[asc|desc], col2[asc|desc]...])
```

Description: This function returns the ranking of values. The data in rows with the same column value in ORDER BY has the same ranking.

Parameters:

- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: the value on which the ranking values are based.

Return value: A value of the BIGINT type is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Group employees by department, sort the employees in each group in descending order of sal, and then obtain the rankings of employees in each group.

```
select deptno
       ,ename
       ,sal
       ,dense_rank() over (partition by deptno order by sal desc) as nums
-- DEPTNO (department) is the column in the window that is used for calculation. Values in the sal column are sorted
to generate the ranking of each employee.
from emp;
-- The following result is returned:
+-----+-----+-----+-----+
|deptno |ename |sal   |nums |
+-----+-----+-----+-----+
| 10    |JACCKA|5000.0| 1    |
| 10    |KING  |5000.0| 1    |
| 10    |CLARK |2450.0| 2    |
| 10    |WELAN |2450.0| 2    |
| 10    |TEBAGE|1300.0| 3    |
| 10    |MILLER|1300.0| 3    |
| 20    |SCOTT |3000.0| 1    |
| 20    |FORD  |3000.0| 1    |
| 20    |JONES |2975.0| 2    |
| 20    |ADAMS |1100.0| 3    |
| 20    |SMITH |800.0 | 4    |
| 30    |BLAKE |2850.0| 1    |
| 30    |ALLEN |1600.0| 2    |
| 30    |TURNER|1500.0| 3    |
| 30    |MARTIN|1250.0| 4    |
| 30    |WARD  |1250.0| 4    |
| 30    |JAMES |950.0 | 5    |
+-----+-----+-----+-----+
```

1.6.6.4.11. RANK

This topic describes the RANK function and provides examples of using this function.

Syntax:

```
Bigint rank() over(partition by [col1, col2...]
order by [col1[asc|desc], col2[asc|desc]...])
```

Description: This function determines the ranking of values in rows with the same column value, after the values are sorted in descending order by using the ORDER BY clause.

Parameters:

- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: the field based on which values are ranked.

Return value: A value of the BIGINT type is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Group employees by department, sort the employees in each group in descending order of sal, and then obtain the ranking values of employees in each group.

```
select deptno
       , ename
       , sal
       , rank() over (partition by deptno order by sal desc) as nums
-- DEPTNO (department) is the column in the window that is used for calculation. Values in the sal column are sorted
to generate the ranking value for each employee.
from emp;
-- The following result is returned:
+-----+-----+-----+-----+
|deptno | ename | sal   | nums |
+-----+-----+-----+-----+
| 10    | JACCKA | 5000.0 | 1    |
| 10    | KING   | 5000.0 | 1    |
| 10    | CLARK  | 2450.0 | 3    |
| 10    | WELAN  | 2450.0 | 3    |
| 10    | TEBAGE | 1300.0 | 5    |
| 10    | MILLER | 1300.0 | 5    |
| 20    | SCOTT  | 3000.0 | 1    |
| 20    | FORD   | 3000.0 | 1    |
| 20    | JONES  | 2975.0 | 3    |
| 20    | ADAMS  | 1100.0 | 4    |
| 20    | SMITH  | 800.0  | 5    |
| 30    | BLAKE  | 2850.0 | 1    |
| 30    | ALLEN  | 1600.0 | 2    |
| 30    | TURNER | 1500.0 | 3    |
| 30    | MARTIN | 1250.0 | 4    |
| 30    | WARD   | 1250.0 | 4    |
| 30    | JAMES  | 950.0  | 6    |
+-----+-----+-----+-----+
```

1.6.6.4.12. LAG

This topic describes the LAG function and provides examples of using this function.

Syntax:

```
lag(expr, Bigint offset, default) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]])
```

Description: This function returns the values for a row at a given offset before the current row. For example, if the current row is m , the values for the $m - \text{offset}$ th row are returned.

Parameters:

- `expr`: a value of any data type.
- `offset`: the offset, which is a constant of the BIGINT type. The value of the offset must be greater than 0. If the input value is of the STRING or DOUBLE type, it is implicitly converted into the BIGINT type before calculation.
- `default`: The default value when the offset is out of the valid range. The value of this parameter must be a constant. The default value of this parameter is NULL.
- `partition by [col1, col2...]`: the columns in the window that is used for calculation.
- `order by col1[asc|desc], col2[asc|desc]`: specifies how values in the rows are sorted.

Return value: A value of the same data type as `expr` is returned.

Examples:

```
select seq, lag(seq+100, 1) over (partition by window order by seq) as r from sliding_window;  
-- The following result is returned:
```

```
+-----+-----+  
| seq  | r    |  
+-----+-----+  
| 0    | NULL |  
| 1    | 100  |  
| 2    | 101  |  
| 3    | 102  |  
| 4    | 103  |  
| 5    | 104  |  
| 6    | 105  |  
| 7    | 106  |  
| 8    | 107  |  
| 9    | 108  |  
+-----+-----+
```

1.6.6.4.13. LEAD

This topic describes the LEAD function and provides examples of using this function.

Syntax:

```
lead(expr, Bigint offset, default) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]])
```

Description: This function returns the values for a row at a given offset after the current row. For example, if the current row is m , the values for the $m + \text{offset}$ th row are returned.

Parameters:

- `expr`: a value of any data type.
- `offset`: the offset, which is a constant of the BIGINT type. The value of this parameter must be greater than

0. If the input value is of the STRING or DOUBLE type, it is implicitly converted into the BIGINT type.
- default: the default value when the value of offset is out of the valid range. The value of this parameter is a constant. The default value of this parameter is NULL.
 - partition by [col1, col2...]: the columns in the window that is used for calculation.
 - order by [col1[asc|desc], col2[asc|desc]: specifies how return values are sorted.

Return value: A value of the same data type as expr is returned.

Examples:

Column information in the test_lead table:

c_int_a,c_Double_a,c_String_a,c_String_b,c_time_a,c_time_b,c_String_in_fact_num.

```
select c_double_a,c_string_b,c_int_a,lead(c_int_a,1) over(partition by c_double_a order by c_string_b) from test_lead;
select c_string_a,c_time_b,c_double_a,lead(c_double_a,1) over(partition by c_string_a order by c_time_b) from test_lead;
select c_string_in_fact_num,c_string_a,c_int_a,lead(c_int_a) over(partition by c_string_in_fact_num order by c_string_a) from test_lead;
```

1.6.6.4.14. PERCENT_RANK

This topic describes the PERCENT_RANK function.

Syntax:

```
percent_rank() over(partition by [col1, col2...]
order by [col1[asc|desc], col2[asc|desc]...])
```

Description: This function returns the relative percent rank of a row in a group of data.

Parameters:

- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by col1[asc|desc], col2[asc|desc]: the value that is used to calculate the percent rank.

Return value: A value of the DOUBLE type is returned. Valid values: [0,1]. The relative percent rank is calculated by using the following formula: $(\text{Rank} - 1) / (\text{Number of rows} - 1)$

 **Note** The number of rows in a single window cannot exceed 10 million.

1.6.6.4.15. ROW_NUMBER

This topic describes the ROW_NUMBER function and provides examples of using this function.

Syntax:

```
row_number() over(partition by [col1, col2...]
order by [col1[asc|desc], col2[asc|desc]...])
```

Description: This function returns the ordinal number of the current row within a group of rows, counting from 1.

Parameters:

- partition by [col1, col2...]: the columns in the window that is used for calculation.

- order by [col1[asc|desc], col2[asc|desc]: the values that need to be sorted to return the ordinal number of the current row.

Return value: A value of the BIGINT type is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Group employees by department, sort the employees in each group in descending order of sal, and then obtain the ordinal numbers of employees in each group.

```

select deptno
       ,ename
       ,sal
       ,row_number() over (partition by deptno order by sal desc) as nums
-- DEPTNO (department) is the column in the window that is used for calculation. Values in the sal column are sorted
to return the ordinal number of the current row.
  from emp;
-- The following result is returned:
+-----+-----+-----+-----+
|deptno |ename |sal   |nums |
+-----+-----+-----+-----+
| 10    |JACCKA|5000.0| 1    |
| 10    |KING  |5000.0| 2    |
| 10    |CLARK |2450.0| 3    |
| 10    |WELAN |2450.0| 4    |
| 10    |TEBAGE|1300.0| 5    |
| 10    |MILLER|1300.0| 6    |
| 20    |SCOTT |3000.0| 1    |
| 20    |FORD  |3000.0| 2    |
| 20    |JONES |2975.0| 3    |
| 20    |ADAMS |1100.0| 4    |
| 20    |SMITH |800.0  | 5    |
| 30    |BLAKE |2850.0| 1    |
| 30    |ALLEN |1600.0| 2    |
| 30    |TURNER|1500.0| 3    |
| 30    |MARTIN|1250.0| 4    |
| 30    |WARD  |1250.0| 5    |
| 30    |JAMES |950.0  | 6    |
+-----+-----+-----+-----+

```

1.6.6.4.16. CLUSTER_SAMPLE

This topic describes the CLUSTER_SAMPLE function and provides examples of using this function.

Syntax:

```

boolean cluster_sample([Bigint x, Bigint y])
over(partition by [col1, col2..])

```

Description: This function performs cluster sampling.

Parameters:

- **x**: a constant of the BIGINT type. The value of this parameter must be greater than or equal to 1. If y is specified, x indicates that a window is divided into x portions. Otherwise, x indicates that the records of x rows in a window are extracted. In this case, True is returned for the x rows. If the value of x is NULL, NULL is returned.
- **y**: a constant of the BIGINT type. The value of y must be greater than or equal to 1 and less than or equal to x. y indicates that y records of the x portions in a window are extracted. In this case, True is returned for the y records. If the value of y is NULL, NULL is returned.
- **partition by [col1, col2...]**: the columns in the window that is used for calculation.

Return value: A value of the BOOLEAN type is returned.

Examples:

The test_tbl table contains two columns: key and value. key specifies a group, which can be groupa or groupb. value indicates the value of key. Data in the test_tbl table:

```
+-----+-----+
| key | value |
+-----+-----+
| groupa | -1.34764165478145 |
| groupa | 0.740212609046718 |
| groupa | 0.167537127858695 |
| groupa | 0.630314566185241 |
| groupa | 0.0112401388646925 |
| groupa | 0.199165745875297 |
| groupa | -0.320543343353587 |
| groupa | -0.273930924365012 |
| groupa | 0.386177958942063 |
| groupa | -1.09209976687047 |
| groupb | -1.10847690938643 |
| groupb | -0.725703978381499 |
| groupb | 1.05064697475759 |
| groupb | 0.135751224393789 |
| groupb | 2.13313102040396 |
| groupb | -1.11828960785008 |
| groupb | -0.849235511508911 |
| groupb | 1.27913806620453 |
| groupb | -0.330817716670401 |
| groupb | -0.300156896191195 |
| groupb | 2.4704244205196 |
| groupb | -1.28051882084434 |
+-----+-----+
```

If you want to extract a sample of 10% of the values from each group, execute the following MaxCompute SQL statement:

```
select key, value from (select key, value, cluster_sample(10, 1) over(partition by key) as flag from tbl) sub where flag = true;
-- The following result is returned:
+-----+-----+
| key | value |
+-----+-----+
| groupa | -0.273930924365012 |
| groupb | -1.11828960785008 |
+-----+-----+
```

1.6.6.4.17. NTILE

This topic describes the NTILE function and provides examples of using this function.

Syntax:

```
BIGINT ntile(BIGINT n) over(partition by [col1, col2...]
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function divides rows into n ranking groups of as equal size as possible and returns the ranking group that a given row falls into. If rows are not evenly divided into ranking groups, more rows are included in the first ranking group.

Parameters: n: a value of the BIGINT type.

Return value: A value of the BIGINT type is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Classify all employees into three groups by department, sort employees in each group in descending order of sal, and then obtain the ranking groups of employees in each group.

```
select deptno,ename,sal,ntile(3) over(partition by deptno order by sal desc) as nt3 from emp;
-- The following result is returned:
```

```
+-----+-----+-----+-----+
|deptno | ename | sal  | nt3  |
+-----+-----+-----+-----+
| 10    | JACCKA | 5000.0 | 1    |
| 10    | KING  | 5000.0 | 1    |
| 10    | WELAN | 2450.0 | 2    |
| 10    | CLARK | 2450.0 | 2    |
| 10    | TEBAGE | 1300.0 | 3    |
| 10    | MILLER | 1300.0 | 3    |
| 20    | SCOTT | 3000.0 | 1    |
| 20    | FORD  | 3000.0 | 1    |
| 20    | JONES | 2975.0 | 2    |
| 20    | ADAMS | 1100.0 | 2    |
| 20    | SMITH | 800.0  | 3    |
| 30    | BLAKE | 2850.0 | 1    |
| 30    | ALLEN | 1600.0 | 1    |
| 30    | TURNER | 1500.0 | 2    |
| 30    | MARTIN | 1250.0 | 2    |
| 30    | WARD  | 1250.0 | 3    |
| 30    | JAMES | 950.0  | 3    |
+-----+-----+-----+-----+
```

1.6.6.4.18. NTH_VALUE

This topic describes the NTH_VALUE function and provides examples of using this function.

Syntax:

```
nth_value(expr, number [, skipNull]) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function calculates the nth value in the window. If the value of n exceeds the total number of rows in the window, NULL is returned.

Parameters:

- expr: a value of any basic data type.
- number: an integer that is greater than or equal to 1.
- skipNull: specifies whether to ignore the rows with NULL values when you calculate the nth value. The value of this parameter is of the BOOLEAN type. The default value is False.
- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: If ORDER BY is not specified, the value of expr of the nth row in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the value of expr of the nth row from the starting row to the current row in the current window is returned.

Return value: A value of the same data type as expr is returned.

Examples:

```
select user_id, price, nth_value(price, 2) over  
  (partition by user_id) as nth_value from test_src;  
+-----+-----+-----+  
| user_id | price | nth_value |  
+-----+-----+-----+  
| 1 | 5.5 | 4.5 |  
| 1 | 4.5 | 4.5 | -- This row is the second row in the current window.  
| 1 | 6.5 | 4.5 |  
| 1 | 5.5 | 4.5 |  
| 2 | NULL | 3.0 |  
| 2 | 3.0 | 3.0 | -- This row is the second row in the current window.  
| 3 | 4.0 | NULL |  
| 3 | NULL | NULL | -- This row is the second row in the current window.  
+-----+-----+-----+  
-- If ORDER BY is not specified, rows from the first row to the last row are in the current window. The value of the second row is returned.  
select user_id, price, nth_value(price, 3) over  
  (partition by user_id) as nth_value from test_src;  
+-----+-----+-----+  
| user_id | price | nth_value |  
+-----+-----+-----+  
| 1 | 5.5 | 6.5 |  
| 1 | 4.5 | 6.5 |  
| 1 | 6.5 | 6.5 | -- This row is the third row in the current window.  
| 1 | 5.5 | 6.5 |  
| 2 | NULL | NULL |  
| 2 | 3.0 | NULL |  
| 3 | 4.0 | NULL |  
| 3 | NULL | NULL |  
+-----+-----+-----+  
-- If ORDER BY is not specified, rows from the first row to the last row are in the current window. The value of the third row is returned.  
-- The second and third windows have only two rows.  
select user_id, price, nth_value(price, 2) over
```

```
select user_id, price, nth_value(price, 2) over
(partition by user_id order by price) as nth_value from test_src;
```

user_id	price	nth_value
1	4.5	NULL
1	5.5	5.5
1	5.5	5.5
1	6.5	5.5
2	NULL	NULL
2	3.0	3.0
3	NULL	NULL
3	4.0	4.0

-- If ORDER BY is specified, rows from the first row to the current row belong to the current window. The value of the second row is returned.

```
select user_id, price, nth_value(price, 1, true) over
(partition by user_id) as nth_value from test_src;
```

user_id	price	nth_value
1	5.5	5.5
1	4.5	5.5
1	6.5	5.5
1	5.5	5.5
2	NULL	3.0
2	3.0	3.0
3	4.0	4.0
3	NULL	4.0

-- If ORDER BY is not specified, rows from the first row to the last row belong to the current window.
-- The value of the first row is returned. skipNull is set to True.

1.6.6.4.19. CUME_DIST

This topic describes the CUME_DIST function and provides examples of using this function.

Syntax:

```
cume_dist() over(partition by [col1, col2...]  
order by [col1[asc|desc], col2[asc|desc]...])
```

Description: This function calculates the cumulative distribution. The cumulative distribution is the ratio of rows whose values are greater than or equal to the current value to all rows in a group.

Parameters: ORDER BY: the value used for comparison.

Return value: The ratio of the number of rows whose values are greater than or equal to the current value in the group to the total number of rows in the group is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Group all employees by department and obtain the cumulative distribution of sal for each group.

```
select deptno
, ename
, sal
, concat(round(cume_dist() over(partition by deptno order by sal desc)*100,2),'%') as cume_dist
from emp;
```

-- The following result is returned:

```
+-----+-----+-----+-----+
|deptno | ename | sal   | cume_dist |
+-----+-----+-----+-----+
| 10    | JACCKA | 5000.0 | 33.33% |
| 10    | KING  | 5000.0 | 33.33% |
| 10    | CLARK | 2450.0 | 66.67% |
| 10    | WELAN | 2450.0 | 66.67% |
| 10    | TEBAGE | 1300.0 | 100.0% |
| 10    | MILLER | 1300.0 | 100.0% |
| 20    | SCOTT | 3000.0 | 40.0% |
| 20    | FORD  | 3000.0 | 40.0% |
| 20    | JONES | 2975.0 | 60.0% |
| 20    | ADAMS | 1100.0 | 80.0% |
| 20    | SMITH | 800.0  | 100.0% |
| 30    | BLAKE | 2850.0 | 16.67% |
| 30    | ALLEN | 1600.0 | 33.33% |
| 30    | TURNER | 1500.0 | 50.0% |
| 30    | MARTIN | 1250.0 | 83.33% |
| 30    | WARD  | 1250.0 | 83.33% |
| 30    | JAMES | 950.0  | 100.0% |
+-----+-----+-----+-----+
```

1.6.6.4.20. FIRST_VALUE

This topic describes the FIRST_VALUE function and provides examples of using this function.

Syntax:

```
first_value(expr) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function sorts rows and returns the first value in the range from the starting row to the current row.

Parameters:

- expr: a value of any basic data type.
- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: If ORDER BY is not specified, the value of expr of the starting row in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the value of expr of the starting row in the current window is returned.

Return value: A value of the same data type as expr is returned.

Examples:

```
select user_id, price, first_value(price) over  
(partition by user_id) as first_value from test_src;  
+-----+-----+-----+  
| user_id | price | first_value |  
+-----+-----+-----+  
| 1 | 5.5 | 5.5 | -- This row is the starting row of this window.  
| 1 | 4.5 | 5.5 |  
| 1 | 6.5 | 5.5 |  
| 1 | 5.5 | 5.5 |  
| 2 | NULL | NULL | -- This row is the starting row of this window.  
| 2 | 3.0 | NULL |  
| 3 | 4.0 | 4.0 | -- This row is the starting row of this window.  
| 3 | NULL | 4.0 |  
+-----+-----+-----+  
-- If ORDER BY is not specified, rows from the first row to the last row belong to the current window. The value of the  
starting row in the current window is returned.  
select user_id, price, first_value(price) over  
(partition by user_id order by price) as first_value from test_src;  
+-----+-----+-----+  
| user_id | price | first_value |  
+-----+-----+-----+  
| 1 | 4.5 | 4.5 | -- This row is the starting row of this window.  
| 1 | 5.5 | 4.5 |  
| 1 | 5.5 | 4.5 |  
| 1 | 6.5 | 4.5 |  
| 2 | NULL | NULL | -- This row is the starting row of this window.  
| 2 | 3.0 | NULL |  
| 3 | NULL | NULL | -- This row is the starting row of this window.  
| 3 | 4.0 | NULL |  
+-----+-----+-----+  
-- If ORDER BY is specified, rows from the first row to the current row belong to the current window. The value of the  
starting row in the current window is returned.
```

1.6.6.4.21. LAST_VALUE

This topic describes the LAST_VALUE function and provides examples of using this function.

Syntax:

```
last_value(expr) over(partition by [col1, col2...]  
[order by [col1[asc|desc], col2[asc|desc]...]] [windowing_clause])
```

Description: This function sorts rows and returns the last value in the range from the starting row to the current row.

Parameters:

- expr: a value of any basic data type.
- partition by [col1, col2...]: the columns in the window that is used for calculation.
- order by [col1[asc|desc], col2[asc|desc]: If ORDER BY is not specified, the value of expr of the last row in the current window is returned. If ORDER BY is specified, the returned results are sorted in the specified order and the value of expr of the current row in the current window is returned.

Return value: A value of the same data type as expr is returned.

Examples:

```
select user_id, price, last_value(price) over  
(partition by user_id) as last_value from test_src;  
+-----+-----+-----+  
|user_id |price  |last_value |  
+-----+-----+-----+  
| 1      |5.5    |5.5        |  
| 1      |4.5    |5.5        |  
| 1      |6.5    |5.5        |  
| 1      |5.5    |5.5        |-- This row is the last row in this window.  
| 2      |NULL   |3.0        |  
| 2      |3.0    |3.0        |-- This row is the last row in this window.  
| 3      |4.0    |NULL       |  
| 3      |NULL   |NULL       |-- This row is the last row in this window.  
+-----+-----+-----+  
-- If ORDER BY is not specified, the rows from the first row to the last row belong to the current window, and the value of the last row in the current window is returned.  
select user_id, price, last_value(price) over  
(partition by user_id order by price) as last_value from test_src;  
+-----+-----+-----+  
|user_id |price  |last_value |  
+-----+-----+-----+  
| 1      |4.5    |4.5        |-- This row is the current row in the current window.  
| 1      |5.5    |5.5        |-- This row is the current row in the current window.  
| 1      |5.5    |5.5        |-- This row is the current row in the current window.  
| 1      |6.5    |6.5        |-- This row is the current row in the current window.  
| 2      |NULL   |NULL       |-- This row is the current row in the current window.  
| 2      |3.0    |3.0        |-- This row is the current row in the current window.  
| 3      |NULL   |NULL       |-- This row is the current row in the current window.  
| 3      |4.0    |4.0        |-- This row is the current row in the current window.  
+-----+-----+-----+  
-- If ORDER BY is specified, rows from the first row to the current row belong to the current window. The value of the current row in the current window is returned.
```

1.6.6.5. Aggregate functions

1.6.6.5.1. Expressions of filter conditions in aggregate functions

This topic describes the expressions of filter conditions in aggregate functions and provides examples.

Filter conditions can be added for all aggregate functions. If FILTER is specified, only the rows for which the value of where_condition is TRUE are processed by the required aggregate functions.

Syntax:

```
aggregate_name(expression[,...]) [FILTER (WHERE where_condition)]
```

Examples:

```
SELECT
  SUM(x),
  SUM(x) FILTER (WHERE y > 1),
  SUM(x) FILTER (WHERE y > 2)
FROM VALUES(NULL, 1),(1,2),(2,3),(3,NULL) AS t(x,y);
```

The following result is returned:

```
+-----+-----+-----+
|_c0  |_c1  |_c2  |
+-----+-----+-----+
| 6   | 3   | 2   |
+-----+-----+-----+
```

Note

- Only built-in aggregate functions support FILTER (WHERE where_condition). User-defined aggregate functions (UDAFs) do not support FILTER (WHERE where_condition).
- COUNT(*) cannot be used with FILTER (WHERE where_condition). To implement this feature, use the COUNT_IF function.

1.6.6.5.2. COUNT

This topic describes the COUNT function and provides examples of using this function.

Syntax:

```
bigint count([distinct|all] value)
```

Description: This function returns the number of records.

Parameters:

- distinct|all: specifies whether to deduplicate records during counting. The default value is all, which indicates that all records are counted. If this parameter is set to distinct, only records with distinct values are counted.
- value: a value of any data type. If the input value for a row is NULL, this row is not used for calculation. You can set value to an asterisk (*). This means that you can run count(*) to count all records.

Return value: A value of the BIGINT type is returned.

Examples:

The tbla table contains the col1 column of the BIGINT type. Data in the tbla table:

```
+-----+
| col1 |
+-----+
| 1 |
+-----+
| 2 |
+-----+
| NULL |
+-----+
```

Execute the following SQL statements:

```
select count(*) from tbla;
-- 3 is returned.
select count(col1) from tbla;
-- 2 is returned.
```

The COUNT function can be used with the GROUP BY clause. For example, the test_src table contains two columns: key and value. key is of the STRING type and value is of the DOUBLE type.

Data in the test_src table:

```
+-----+-----+
| key | value |
+-----+-----+
| a | 2.0 |
+-----+-----+
| a | 4.0 |
+-----+-----+
| b | 1.0 |
+-----+-----+
| b | 3.0 |
+-----+-----+
```

Execute the following statement:

```
select key, count(value) as count from test_src group by key;
-- The following result is returned:
+-----+-----+
| key | count |
+-----+-----+
| a | 2 |
+-----+-----+
| b | 2 |
+-----+-----+
```

The COUNT function aggregates the values with the same key. The usage of other aggregate functions is the same as that of this function and is not described in detail in this document.

1.6.6.5.3. COUNT_IF

This topic describes the aggregate function COUNT_IF and provides examples.

Syntax:

```
bigint count_if(BOOLEAN expr)
```

Description: This function calculates the number of records whose value of expr is TRUE.

Parameters: expr, which is of the BOOLEAN type. If the value for a row is FALSE or NULL, the row is not counted.

Return value: A value of the BIGINT type is returned.

Examples:

```
SELECT COUNT_IF(x > 1), COUNT_IF(x <= 1) FROM VALUES(NULL),(0),(1),(2) AS t(x);
```

The following result is returned:

```
+-----+-----+
|_c0  |_c1  |
+-----+-----+
| 1   | 2   |
+-----+-----+
```

1.6.6.5.4. AVG

This topic describes the AVG function and provides examples of using this function.

Syntax:

```
double avg(double value)
decimal avg(decimal value)
```

Description: This function returns the average value of a column.

Parameters: value: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned. If the input value for a row is NULL, this row is not used for calculation. Values of the BOOLEAN type cannot be used for calculation.

Return value: If the input value is of the DECIMAL type, a value of the DECIMAL type is returned. If the input value is not of the DECIMAL type, a value of the DOUBLE type is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value |
+-----+
| 1     |
| 2     |
| NULL  |
+-----+
```

Execute the following statement:

```
select avg(value) as avg from tbla;
-- The average value of the value column is 1.5, which is calculated by using the following formula: (1 + 2)/2
+-----+
| avg |
+-----+
| 1.5 |
+-----+
```

1.6.6.5.5. MAX

This topic describes the MAX function and provides examples of using this function.

Syntax:

```
max(value)
```

Description: This function calculates the maximum value of values in a column.

Parameters: value: a value of any data type. If the value for a row is NULL, this row that corresponds to the column is not used for calculation. Values of the BOOLEAN type are not used for calculation.

Return value: A value of the same data type as value is returned.

Examples:

The tbla table contains the col1 column of the BIGINT type. Data in the tbla table:

```
+-----+
| col1 |
+-----+
| 1 |
+-----+
| 2 |
+-----+
| NULL |
+-----+
```

Execute the following statement:

```
select max(value) from tbla;
-- 2 is returned.
```

1.6.6.5.6. MIN

This topic describes the MIN function and provides examples of using this function.

Syntax:

```
min(value)
```

Description: This function calculates the minimum value of values in a column.

Parameters: value: a value of any data type. If the value for a row is NULL, this row that corresponds to the column is not used for calculation. Values of the BOOLEAN type are not used for calculation.

Return value: A value of the same data type as value is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value|
+-----+
| 1 |
+-----+
| 2 |
+-----+
| NULL |
+-----+
```

Execute the following statement :

```
select min(value) from tbla;
-- 1 is returned.
```

1.6.6.5.7. MEDIAN

This topic describes the MEDIAN function and provides examples of using this function.

Syntax:

```
double median(double number)
decimal median(decimal number)
```

Description: This function calculates the median value for a group of values.

Parameters: value: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value|
+-----+
| 1 |
+-----+
| 2 |
+-----+
| 3 |
+-----+
| 4 |
+-----+
| 5 |
+-----+
```

Execute the following statement :

```
select median(value) from tbla;
-- 3.0 is returned.
```

1.6.6.5.8. STDDEV

This topic describes the STDDEV function and provides examples of using this function.

Syntax:

```
double stddev(double number)
decimal stddev(decimal number)
```

Description: This function calculates the population standard deviation.

Parameters: value: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value|
+-----+
| 1 |
+-----+
| 2 |
+-----+
| 3 |
+-----+
| 4 |
+-----+
| 5 |
+-----+
```

Execute the following statement :

```
select stddev(value) from tbla;
-- 1.4142135623730951 is returned.
```

1.6.6.5.9. STDDEV_SAMP

This topic describes the STDDEV_SAMP function and provides examples of using this function.

Syntax:

```
double stddev_samp(double number)
decimal stddev_samp(decimal number)
```

Description: This function returns the sample standard deviation of a group of values.

Parameters: value: a value of the DOUBLE or DECIMAL type. If the input value is of the STRING or BIGINT type, it is implicitly converted into the DOUBLE type before calculation. If the input value is not of the STRING or BIGINT type, an error is returned.

Return value: A value of the DOUBLE or DECIMAL type is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value|
+-----+
| 1 |
+-----+
| 2 |
+-----+
| 3 |
+-----+
| 4 |
+-----+
| 5 |
+-----+
```

Execute the following statement:

```
select stddev_samp(value) from tbla;
-- 1.58113883008418981 is returned.
```

1.6.6.5.10. SUM

This topic describes the SUM function and provides examples of using this function.

Syntax:

```
sum(value)
```

Description: This function returns the sum of values in specified rows.

Parameters: value: a value of the DOUBLE, DECIMAL, or BIGINT type. If the input value is of the STRING type, it is implicitly converted into the DOUBLE type before calculation. If the input value for a row is NULL, this row is not used for calculation. Values of the BOOLEAN type are not used for calculation.

Return value: If the input value is of the BIGINT type, a value of the BIGINT type is returned. If the input value is of the DOUBLE or STRING type, a value of the DOUBLE type is returned.

Examples:

The tbla table contains the value column of the BIGINT type. Data in the tbla table:

```
+-----+
| value|
+-----+
| 1 |
+-----+
| 2 |
+-----+
| NULL|
+-----+
```

Execute the following statement :

```
select sum(value) from tbla;
-- 3 is returned.
```

1.6.6.5.11. WM_CONCAT

This topic describes the WM_CONCAT function and provides examples of using this function.

Syntax:

```
string wm_concat(string separator, string str)
```

Description: This function returns a string of values that are separated by using a given delimiter. The delimiter is specified by separator.

Parameters:

- separator: the delimiter, which is a constant of the STRING type. If it is not of the STRING type or is not a constant, an error is returned.
- str: a value of the STRING type. If the input value is of the BIGINT, DOUBLE, or DATETIME type, it is implicitly converted into the STRING type before calculation. If the input value is not of the BIGINT, DOUBLE, or DATETIME type, an error is returned.

Return value: A value of the STRING type is returned.

 **Note** If test_src in select wm_concat(',', name) from test_src; is an empty set, NULL is returned.

Examples:

Group and sort values in the test table and use a string to list the values in the same group.

```
-- Create the test table.
create table test(id int , alphabet string);
-- Insert data into the test table.
insert into test values (1,'a'),(1,'b'),(1,'c'),(2,'D'),(2,'E'),(2,'F');
-- Group and sort values in the test table based on the id column. Then, use a string to list values in the same group.
select id,wm_concat(',',alphabet) from test group by id order by id limit 100;
+-----+-----+
| id  | _c1  |
+-----+-----+
| 1   | abc  |
| 2   | DEF  |
+-----+-----+
```

1.6.6.5.12. PERCENTILE

This topic describes the PERCENTILE function and provides examples of using this function.

Syntax:

```
double percentile(bigint col, p)
array<double> percentile(bigint col, array(p1 [, p2]...))
```

Description: This function returns the pth percentile of the specified column. p must be between 0 and 1.

 **Notice** You can calculate the percentile only for integer values.

Parameters:

- col: the name of the table column of the BIGINT type.
- p: the percentile that must be between 0 and 1.

Examples:

The var_test table contains the c1 column. Data in the var_test table:

```
+-----+
| c1 |
+-----+
| 8 |
| 9 |
| 10 |
| 11 |
+-----+
```

Execute the following statement to calculate the pth percentile of the c1 column.

```
select percentile(c1,0),percentile(c1,0.3),percentile(c1,0.5),percentile(c1,1) from var_test;
-- The following result is returned:
+-----+-----+-----+-----+
| _c0 | _c1 | _c2 | _c3 |
+-----+-----+-----+-----+
| 8.0 | 8.9 | 9.5 | 11.0 |
+-----+-----+-----+-----+
select percentile(c1,array(0,0.3,0.5,1))from var_test;
-- The following result is returned:
+-----+
| _c0 |
+-----+
| [8, 8.9, 9.5, 11] |
+-----+
```

1.6.6.5.13. Additional aggregate functions

1.6.6.5.13.1. Usage notes

This topic describes the configuration operation that you must perform before you use additional aggregate functions.

MaxCompute V2.0 provides more aggregate functions. You must add the following SET statement before the SQL statements for these new functions:

```
set odps.sql.type.system.odps2=true;
```

 **Note** You must submit and execute the SET statement and the SQL statements for the new functions at the same time.

1.6.6.5.13.2. COLLECT_LIST

This topic describes the COLLECT_LIST function and provides examples of using this function.

Syntax:

```
ARRAY collect_list(col)
```

Description: This function aggregates the expressions that are specified by col in a given group into an array.

Parameters: col: a column of the table, which is of any data type.

Return value: A value of the ARRAY type is returned.

1.6.6.5.13.3. COLLECT_SET

This topic describes the COLLECT_SET function.

Syntax:

```
array collect_set(col)
```

Description: This function aggregates the expressions that are specified by col in a given group into an array without duplicate elements.

Parameters: col: a table column, which can be of any data type.

Return value: A value of the ARRAY type is returned.

1.6.6.5.13.4. VARIANCE or VAR_POP

This topic describes the VARIANCE or VAR_POP function and provides examples of using this function.

Syntax:

```
DOUBLE variance(col)  
DOUBLE var_pop(col)
```

Description: This function calculates the variance of a specified column of a numeric data type.

Parameters: col: the column of a numeric data type. If it is not of a numeric data type, NULL is returned.

Return value: A value of the DOUBLE type is returned.

Examples:

The test table contains the c1 column. Data in the test table:

```
+-----+
| c1 |
+-----+
| 8 |
| 9 |
| 10 |
| 11 |
+-----+
```

Execute the following statement to calculate the variance of the c1 column:

```
select variance(c1) from test;
-- You can also execute the following statement:
select var_pop(c1) from test;
-- The following result is returned:
+-----+
| _c0 |
+-----+
| 1.25 |
+-----+
```

1.6.6.5.13.5. VAR_SAMP

This topic describes the VAR_SAMP function and provides examples of using this function.

Syntax:

```
DOUBLE var_samp(col)
```

Description: This function calculates the sample variance of a specified column of a numeric data type.

Parameters: col: a column of a numeric data type. If it is not of a numeric data type, NULL is returned.

Return value: A value of the DOUBLE type is returned

Examples:

The test table contains the c1 column. Data in the test table:

```
+-----+
| c1 |
+-----+
| 8 |
| 9 |
| 10 |
| 11 |
+-----+
```

Execute the following statement to calculate the sample variance of the c1 column:

```
select var_samp(c1) from test;
-- The following result is returned:
+-----+
|_c0|
+-----+
|1.666666666666667|
+-----+
```

1.6.6.5.13.6. COVAR_POP

This topic describes the COVAR_POP function and provides examples of using this function.

Syntax:

```
DOUBLE covar_pop(col1, col2)
```

Description: This function calculates the population covariance of two columns of a numeric data type.

Parameters: col1 and col2: the columns of a numeric data type. If they are not of a numeric data type, NULL is returned.

Return value: A value of the DOUBLE type is returned.

Examples:

The test table contains the c1 and c2 columns. Data in the test table:

```
+-----+-----+
|c1|c2|
+-----+-----+
|3|2|
|14|5|
|50|14|
|26|75|
+-----+-----+
```

Execute the following statement to calculate the population covariance of the c1 and c2 columns:

```
select covar_pop(c1,c2) from test;
-- The following result is returned:
+-----+
|_c0|
+-----+
|123.49999999999997|
+-----+
```

1.6.6.5.13.7. COVAR_SAMP

This topic describes the COVAR_SAMP function and provides examples of using this function.

Syntax:

```
DOUBLE covar_samp(col1, col2)
```

Description: This function calculates the sample covariance of two columns of a numeric data type.

Parameters: col1 and col2: the columns of a numeric data type. If they are not of a numeric data type, NULL is returned.

Return value: A value of the DOUBLE type is returned

Examples:

The test table contains the c1 and c2 columns. Data in the test table:

```
+-----+-----+
| c1  | c2  |
+-----+-----+
| 3   | 2   |
| 14  | 5   |
| 50  | 14  |
| 26  | 75  |
+-----+-----+
```

Execute the following statement to calculate the sample covariance of the c1 and c2 columns:

```
select covar_samp(c1,c2) from test;
-- The following result is returned:
+-----+
| _c0  |
+-----+
| 164.66666666666663|
+-----+
```

1.6.6.5.13.8. ANY_VALUE

This topic describes the ANY_VALUE function and provides examples of using this function.

Syntax:

```
ANY_VALUE(value)
```

Description: This function returns a non-deterministic value from a specific column.

Parameters: value: a value of any data type. If the input value for a row is NULL, this row is not used for calculation.

Return value: A value of the same data type as the input value is returned.

Examples:

```
select key, ANY_VALUE(value)
from values (1, 'value1'),(1, 'value2'),(2, 'value3'),(2, NULL),(3, NULL) as t(key, value)
group by key;
-- The following result is returned:
+-----+-----+
| key  | _c1  |
+-----+-----+
| 1    | value1 |
| 2    | value3 |
| 3    | NULL  |
+-----+-----+
```

 **Note** The value returned by ANY_VALUE is not deterministic and different values may be returned for the same input value.

1.6.6.5.13.9. NUMERIC_HISTOGRAM

This topic describes the NUMERIC_HISTOGRAM function.

Syntax:

```
map<double, double> numeric_histogram(bigint buckets, double value)
```

Description: This function returns the approximate histogram of a given column.

Parameters:

- **buckets**: the maximum number of buckets in the column whose approximate histogram is returned. The value must be of the BIGINT type.
- **value**: the column whose approximate histogram you want to obtain. The column must be of the DOUBLE type.

Return value: A value of the MAP<DOUBLE,DOUBLE> type is returned. In the return value, a key indicates an x-coordinate, and its value indicates the height on the approximate histogram of the column.

1.6.6.5.13.10. PERCENTILE_APPROX

This topic describes the PERCENTILE_APPROX function and provides examples of using this function.

Syntax:

```
double percentile_approx(double col, p [, B])  
array<double> percentile_approx(double col, array(p1 [, p2]...) [, B])
```

Description: This function returns the approximate percentile value of the col column at the given percentage p.

Parameters:

- **p**: the given percentile value. Valid values: [0.0,1.0]. You can specify one percentage to return an approximate percentile value or multiple percentages to return an array that consists of percentile values.
- **B**: the accuracy of the return value. A higher accuracy indicates a more accurate value. If you do not specify this parameter, 10000 is used. If the number of values in the col column is less than B, an exact percentile value is returned.

Return value: percentile_approx(double col, p [, B]) returns a single approximate percentile value.
percentile_approx(double col, array(p1 [, p2]...) [, B]) returns an array that consists of multiple percentile values.

Examples:

```
select percentile_approx(10.0, 0.5, 100);  
-- 10.0 is returned.  
select percentile_approx(10.0, array(0.5, 0.4, 0.1), 100);  
-- [10.0,10.0,10.0] is returned.
```

1.6.6.5.13.11. APPROX_DISTINCT

This topic describes the APPROX_DISTINCT function and provides examples of using this function.

Syntax:

```
bigint approx_distinct(value)
```

Description: This function returns the approximate number of distinct input values.

Parameters: value: the input data that is used for deduplication.

Return value: A value of the BIGINT type is returned. If all input values are NULL, 0 is returned. This function produces a standard error of 5%.

1.6.6.6. Other functions

1.6.6.6.1. ARRAY

This topic describes the ARRAY function and provides examples of using this function.

Syntax:

```
ARRAY array(value1,value2, ...)
```

Description: This function creates an array of a given data type that is specified by value.

Parameters: value: any data type. All the values must be of the same data type.

Return value: A value of the ARRAY type is returned.

Examples:

```
select array(123,456,789);  
-- The following result is returned:  
[123, 456, 789]
```

1.6.6.6.2. ARRAY_CONTAINS

This topic describes the ARRAY_CONTAINS function and provides examples of using this function.

Syntax:

```
boolean array_contains(ARRAY<T> a, value v)
```

Description: This function checks whether array a contains value v.

Parameters:

- a: a value of the ARRAY type.
- v: a value of the same data type as other values in the array.

Return value: A value of the BOOLEAN type is returned.

Examples:

```
select array_contains(array('a','b'), 'a');  
-- True is returned.  
select array_contains(array(456,789),123);  
-- False is returned.
```

1.6.6.6.3. CAST

This topic describes the CAST function.

Syntax:

```
cast(expr as <type>)
```

Description: This function converts the result of an expression into another data type. For example, `cast('1' as bigint)` converts '1' of the STRING type into 1 of the INT type. If the conversion fails, an error is returned.

Notes:

- `cast(double as bigint)` : converts a value of the DOUBLE type to the BIGINT type.
- `cast(string as bigint)` : converts a value of the STRING type to the BIGINT type. If the string consists of numerals expressed as integers, it is directly converted into the BIGINT type. If the string consists of numerals expressed in the FLOAT or EXPONENTIAL form, it is converted into the DOUBLE type and then into the BIGINT type.
- The date format for `cast(string as datetime)` and `cast(datetime as > string)` is yyyy-mm-dd hh:mi:ss.

1.6.6.6.4. COALESCE

This topic describes the COALESCE function.

Syntax:

```
coalesce(expr1, expr2, ...)
```

Description: This function returns the first non-NULL value in a list. If all values in the list are NULL, NULL is returned.

Parameters: expr: the values you want to test. All these values must be of the same data type or are all NULL. Otherwise, an error is returned.

 **Note** The list must include at least one parameter. Otherwise, an error is returned.

Return value: A value of the same data type as the expr parameter is returned.

1.6.6.6.5. DECODE

This topic describes the DECODE function and provides examples of using this function.

Syntax:

```
decode(expression, search, result[, search, result]...[, default])
```

Description: This function implements the IF-THEN-ELSE conditional branching feature.

Parameters:

- expression: the expression that you want to compare.
- search: the search item that you want to compare with expression.
- result: the value returned if the values of search and expression match.
- default: the value returned if no search item matches the expression. If default is not specified, NULL is returned. This parameter is optional.

Return value: The matched search value is returned. If no matched item exists, default is returned. If default is not specified, NULL is returned.

Note

- Three or more parameters must be specified.
- All results must be of the same data type or be NULL. Inconsistent data types will cause an error. All values of search and expression must be of the same data type. Otherwise, an error is returned.
- If search in the DECODE function has duplicate values that match the expression, the first mapping result is returned.

Examples:

```
select
decode(customer_id,
1, 'Taobao',
2, 'Alipay',
3, 'Aliyun',
NULL, 'N/A',
'Others') as result
FROM VALUES (1,10),
(1,10),
(2,null),
(3,13),
(4,0),
(null,6),
(5,18)
as t(customer_id,value);
```

In this example, the DECODE function implements the IF-THEN-ELSE statement.

```
if customer_id = 1 then
result := 'Taobao';
elsif customer_id = 2 then
result := 'Alipay';
elsif customer_id = 3 then
result := 'Aliyun';
...
else
result := 'Others';
end if;
```

Note

- In most cases, if "NULL = NULL" appears during data computation, the result returned by the MaxCompute SQL engine is NULL, whereas the DECODE function considers that the two NULL values are the same.
- In this example, if the value of customer_id is NULL, the DECODE function returns N/A.

1.6.6.6.6. EXPLODE

This topic describes the EXPLODE function and provides examples of using this function.

Syntax:

```
explode (var)
```

Description: This function is a user-defined table-valued function (UDTF) that transposes one row into multiple rows.

- If var is of the ARRAY <T> type, the data of the ARRAY type stored in a column is converted into multi-row data.
- If var is of the MAP <K, V> type, each key-value pair of the MAP type stored in a column is transposed to a row with two columns. One column is used for storing keys and the other column is used for storing values.

Parameters: var: a value of the ARRAY <T> or MAP <K, V> type.

Return value: transposed rows.

Note

Limits on the EXPLODE function:

- A SELECT statement can specify only the column that is used for this function. Other columns cannot be specified.
- This function cannot be used with the GROUP BY, CLUSTER BY, DISTRIBUTE BY, or SORT BY clause.

Examples:

```
select explode(array(null, 'a', 'b', 'c'));
-- The following result is returned:
+-----+
| col  |
+-----+
| NULL |
| a    |
| b    |
| c    |
+-----+
```

1.6.6.6.7. GET_IDCARD_AGE

This topic describes the GET_IDCARD_AGE function.

Syntax:

```
get_idcard_age(idcardno)
```

Description: This function returns the current age based on the ID card number. The current age is the current year minus the birth year in the ID card number.

Parameters: idcardno: the ID card number of the STRING type. The value is a 15-digit or 18-digit number. During data computation, the validity of the ID card is verified based on the province code and the last check code. If the verification fails, NULL is returned.

Return value: A value of the BIGINT type is returned. If the input value is NULL, NULL is returned. If the difference between the current year and the birth year is greater than 100, NULL is returned.

1.6.6.6.8. GET_IDCARD_BIRTHDAY

This topic describes the GET_IDCARD_BIRTHDAY function.

Syntax:

```
get_idcard_birthday(idcardno)
```

Description: This function returns the date of birth based on the ID card number.

Parameters: idcardno: the ID card number of the STRING type. The value is a 15-digit or 18-digit number. During data computation, the validity of the ID card is verified based on the province code and the last check code. If the verification fails, NULL is returned.

Return value: A value of the DATETIME type is returned. If the input value is NULL, NULL is returned.

1.6.6.6.9. GET_IDCARD_SEX

This topic describes the GET_IDCARD_SEX function.

Syntax:

```
get_idcard_sex(idcardno)
```

Description: This function returns the gender based on the ID card number. The return value is M or F. M indicates male and F indicates female.

Parameters: idcardno: the ID card number of the STRING type. The value is a 15-digit or 18-digit number. During data computation, the validity of the ID card is verified based on the province code and the last check code. If the verification fails, NULL is returned.

Return value: A value of the STRING type is returned. If the input value is NULL, NULL is returned.

1.6.6.6.10. GREATEST

This topic describes the GREATEST function.

Syntax:

```
greatest(var1, var2, ...)
```

Description: This function returns the maximum value of the input parameters.

Parameters: var1 and var2: values of the BIGINT, DOUBLE, DECIMAL, DATETIME, or STRING type. If all input values are NULL, NULL is returned.

Return value:

- The maximum value in the values of all input parameters. If implicit conversion is not required, the data type of the return value is the same as the data types of input parameters.
- NULL is interpreted as the minimum value.
- If a data type conversion is performed among the DOUBLE, BIGINT, and STRING types, a value of the DOUBLE type is returned. If a data type conversion is performed between the STRING and DATETIME types, a value of the DATETIME type is returned. If a data type conversion is performed among the DECIMAL, DOUBLE, BIGINT, and STRING types, a value of the DECIMAL type is returned. Implicit conversions of other data types are not allowed.
- If `odps.sql.hive.compatible` is set to true and the value of an input parameter is NULL, NULL is returned.

1.6.6.6.11. INDEX

This topic describes the INDEX function and provides examples of using this function.

Syntax:

```
index(var1[var2])
```

Description: This function returns the var2th value if var1 is of the ARRAY<T> type and returns the value whose key is var2 in var1 if var1 is of the MAP<K, V> type.

Parameters:

- var: a value of the ARRAY<T> or MAP<K, V> type.
- var2: If the value of var1 is of the ARRAY<T> type, the value of var2 is of the BIGINT type and greater than or equal to 0. If the value of var1 is of the MAP<K, V> type, the value of var2 is of the K type.

Return value:

- If the value of var1 is of the ARRAY<T> type, a value of the T type is returned. If the value of var2 is beyond the range of elements in the value of the ARRAY<T> type, NULL is returned.
- If the value of var1 is of the MAP<K, V> type, a value of the V type is returned. If the value whose key is var2 does not exist in values of the MAP<K, V> type, NULL is returned.

Examples:

If the value of var1 is an array, execute the following SQL statement:

```
select array('a','b','c')[2] from dual;
-- The following result is returned:
+-----+
| _c0 |
+-----+
| c |
+-----+
```

If the value of var1 is of the MAP<K, V> type, execute the following SQL statement:

```
select str_to_map("test1=1,test2=2")["test1"] from dual;
-- The following result is returned:
+-----+
| _c0 |
+-----+
| 1 |
+-----+
```

Note

- To execute the SQL statement, remove the index, and execute `var1[var2]` directly. Otherwise, a syntax error is returned.
- If the value of var1 is NULL, NULL is returned.

1.6.6.6.12. MAX_PT

This topic describes the MAX_PT function and provides examples of using this function.

Syntax:

```
max_pt(table_full_name)
```

Description: This function returns the maximum value of each first-level partition of a partitioned table in alphabetic order. This function also reads data from the data file in the first-level partitions.

Parameters: `table_full_name`: the table name that has a project name, such as `prj.src`. The value of this parameter is of the `STRING` type. You must have the permissions to read the table.

Return value: The maximum value in each first-level partition is returned.

Examples:

The `tbl` table is a partitioned table and contains the following partitions:

```
pt='20120901'
pt='20120902'
```

The return value of the following statement is `'20120902'`. The MaxCompute SQL statement reads data from the 20120902 partition.

```
select * from tbl where pt=max_pt('myproject.tbl');
```

 **Note** If a partition is added by using the `ALTER TABLE` statement and the partition contains no data file, the maximum value in this partition is not returned.

1.6.6.6.13. ORDINAL

This topic describes the `ORDINAL` function and provides examples of using this function.

Syntax:

```
ordinal(bigint nth, var1, var2, ...)
```

Description: This function sorts the input variables in ascending order, and returns the value in the `nth` bit.

Parameters:

- `nth`: a value of the `BIGINT` type. This parameter specifies the position in which the value is to be returned. If the value of this parameter is `NULL`, `NULL` is returned.
- `var`: the value of the `BIGINT`, `DOUBLE`, `DATETIME`, or `STRING` type.

Return value:

- The value in the `nth` bit is returned. If no implicit conversion is required, the return value is of the same data type as the input parameter.
- If a data type conversion is performed among the `DOUBLE`, `BIGINT`, and `STRING` types, a value of the `DOUBLE` type is returned. If a data type conversion is performed between the `STRING` and `DATETIME` types, a value of the `DATETIME` type is returned. Implicit conversions of other data types are not allowed.
- `NULL` is interpreted as the minimum value.

Examples:

```
ordinal(3, 1, 3, 2, 5, 2, 4, 6) = 2
```

1.6.6.6.14. LEAST

This topic describes the LEAST function.

Syntax:

```
least(var1, var2, ...)
```

Description: This function returns the minimum value of the input parameters.

Parameters: var: a value of the BIGINT, DOUBLE, DECIMAL, DATETIME, or STRING type. If the values of all input parameters are NULL, NULL is returned.

Return value:

- The minimum value of all values of the input parameters is returned. If implicit conversion is not required, the return value must be of the same data type as the input parameters.
- If a data type conversion is performed among the DOUBLE, BIGINT, and STRING types, a value of the DOUBLE type is returned. If a data type conversion is performed between the STRING and DATETIME types, a value of the DATETIME type is returned. If a data type conversion is performed among the DECIMAL, DOUBLE, BIGINT, and STRING types, a value of the DECIMAL type is returned. Implicit conversions of other data types are not allowed.
- NULL is interpreted as the minimum value.

1.6.6.6.15. SIZE

This topic describes the SIZE function and provides examples of using this function.

Syntax:

```
INT size(map)  
INT size(array)
```

Description: size(map) returns the number of key-value pairs in the specified map parameter. size(array) returns the number of elements in the specified array parameter.

Parameters:

- map: a value of the MAP type.
- array: a value of the ARRAY type.

Return value: A value of the INT type is returned.

Examples:

```
select size(map('a',123,'b',456));  
-- 2 is returned.  
select size(map('a',123,'b',456,'c',789));  
-- 3 is returned.  
select size(array('a','b'));  
-- 2 is returned.  
select size(array(123,456,789));  
-- 3 is returned.
```

1.6.6.6.16. SPLIT

This topic describes the SPLIT function and provides examples of using this function.

Syntax:

```
split(str, pat)
```

Description: This function returns an array after str is split with pat.

Parameters:

- str: the string that you want to split, which is of the STRING type.
- pat: a delimiter of the STRING type, which supports regular expressions.

Return value: The result after str is split with pat is returned. The return value is of the ARRAY<STRING> type.

Examples:

```
select split("a,b,c",",");
-- The following result is returned:
+-----+
| _c0 |
+-----+
| [a, b, c] |
+-----+
```

1.6.6.6.17. STR_TO_MAP

This topic describes the STR_TO_MAP function and provides examples of using this function.

Syntax:

```
str_to_map(text [, delimiter1 [, delimiter2]])
```

Description: This function splits text into key-value pairs by using delimiter1 and then splits the key and value in a key-value pair by using delimiter2.

Parameters:

- text: the string that you want to split. The value of this parameter is of the STRING type.
- delimiter1: the delimiter that splits text into key-value pairs. The value of this parameter is of the STRING type. If this parameter is not specified, a comma (,) is used.
- delimiter2: the delimiter that splits each key-value pair into a key and a value. The value of this parameter is of the STRING type. If this parameter is not specified, a period (.) is used.

Return value: The result after text is split by using delimiter1 and delimiter2 is returned. The return value is of the MAP<STRING, STRING> type.

Examples:

```
select str_to_map("test1=1,test2=2");
-- The following result is returned:
+-----+
| a |
+-----+
| {test1:1, test2:2} |
```

1.6.6.6.18. UUID

This topic describes the UUID function.

Syntax:

```
string uuid()
```

Description: This function returns a random ID, for example, 29347a88-1e57-41ae-bb68-a9edbdd94212.

 **Note** The return value of this function is a random global ID that has a low probability of duplication.

1.6.6.6.19. UNIQUE_ID

This topic describes the UNIQUE_ID function.

Syntax:

```
string unique_id()
```

Description: This function returns a unique random ID, for example, 29347a88-1e57-41ae-bb68-a9edbdd94212_1.

 **Note** This function is more efficient than the UUID function.

1.6.6.6.20. SAMPLE

This topic describes the SAMPLE function and provides examples of using this function.

Syntax:

```
boolean sample(x, y, column_name1, column_name2,...)
```

Description: This function samples all values that are read from the column specified by column_name based on x and y, and filters out the rows that do not meet the sampling condition.

Parameters:

- x and y: integer constants that are greater than 0. The values of the two parameters are of the BIGINT type. These parameters indicate that the values fall into x portions by calling the hash function and the yth portion is used. If y is not specified, the first portion is used, and you must not specify column_name. If the value of x or y is not of the BIGINT type or is less than or equal to 0, an error is returned. If the value of y is greater than the value of x, an error is returned. If the value of x or y is NULL, NULL is returned.
- column_name: the column from which you want to sample values. This parameter is optional. If this parameter is not specified, random sampling is performed based on the values of x and y. This parameter can be of any data type, and its value can be NULL. No implicit conversion is performed. If the value of column_name is a constant, an error is returned.

Return value: A value of the BOOLEAN type is returned.

 **Note** To avoid data skew due to NULL values, the system performs uniform hashing on NULL values in the columns specified by `column_name`. This ensures that data evenly falls into `x` portions. If `column_name` is not specified and the amount of data is small, random hashing may be performed. In this case, we recommend that you specify `column_name` for uniform hashing.

Examples:

The `tbla` table contains the `cola` column.

```
select * from tbla where sample (4, 1 , cola) = true;
-- The values in the cola column fall into four portions by calling the hash function, and data in the first portion is extracted.
select * from tbla where sample (4, 2) = true;
-- The values in each row are randomly hashed into four portions, and data in the second portion is extracted.
```

1.6.6.6.21. CASE WHEN expression

This topic describes the CASE WHEN expression and provides examples of using this function.

MaxCompute provides two syntax formats for the CASE WHEN expression.

```
case value
when value1 then result1
when value2 then result2
...
else resultn
end
```

```
case
when (_condition1) then result1
when (_condition2) then result2
when (_condition3) then result3
...
else resultn
end
```

The CASE WHEN expression can return different values based on the calculation result of the value expression. The following example shows that different regions are obtained based on the value of `shop_name`.

```
select
case
when shop_name is null then 'default_region'
when shop_name like 'hang%' then 'zj_region'
end as region
from sale_detail;
```

 Note

- If all result values are of the BIGINT or DOUBLE type, the data types are converted into the DOUBLE type before the values are returned.
- If some result values are of the STRING type, the data types of all values are converted into the STRING type before the values are returned. If the conversion cannot be performed, for example, the BOOLEAN type cannot be converted into the STRING type, an error is returned.
- Conversions between other data types are not allowed.

1.6.6.6.22. IF

This topic describes the IF function and provides examples of using this function.

Syntax:

```
if(testCondition, valueTrue, valueFalseOrNull)
```

Description: This function determines whether testCondition evaluates to true. If testCondition evaluates to true, valueTrue is returned. Otherwise, valueFalse or NULL is returned.

Parameters:

- testCondition: the expression that you want to evaluate. The value of this parameter is of the BOOLEAN type.
- valueTrue: the value returned if testCondition evaluates to true.
- valueFalseOrNull: the value returned if testCondition evaluates to false. It can be set to NULL.

Return value: The data type of the return value is the same as that of valueTrue or valueFalseOrNull.

Examples:

```
select if(1=2,100,200);  
-- The following result is returned:  
+-----+  
|_c0  |  
+-----+  
| 200  |  
+-----+
```

1.6.6.6.23. Additional functions

1.6.6.6.23.1. Usage notes

This topic describes the configuration operation that you must perform before you use other functions that are added to MaxCompute V2.0.

MaxCompute V2.0 provides more other functions. You must add the following SET statement before the SQL statements for these new functions:

```
set odps.sql.type.system.odps2=true;
```

 **Note** You must submit and execute the SET statement and the SQL statements for the new functions at the same time.

1.6.6.6.23.2. MAP

This topic describes the MAP function and provides examples of using this function.

Syntax:

```
map map(K key1, V value1, K key2, V value2, ...)
```

Description: This function creates mappings by using the given key-value pairs.

Parameters:

- **key:** All keys must be of the same data types, such as those after an implicit conversion. The data types must be basic types.
- **value:** All value types, such as those after an implicit conversion, must be the same and can be of any data types.

Return value: A value of the MAP type is returned.

Examples:

The t_table table contains the c1, c2, c3, c4, and c5 columns. The c1, c4, and C5 columns are of the BIGINT type, and the c2 and c3 columns are of the STRING type. Data in the t_table table:

```
+-----+-----+-----+-----+
| c1   | c2 | c3 | c4   | c5   |
+-----+-----+-----+-----+
| 1000 | k11| k21| 86   | 15   |
| 1001 | k12| k22| 97   | 2    |
| 1002 | k13| k23| 99   | 1    |
+-----+-----+-----+-----+
```

Execute the following statement :

```
select map(c2,c4,c3,c5) from t_table;
-- The following result is returned:
+-----+
| _c0 |
+-----+
| {k11:86, k21:15} |
| {k12:97, k22:2} |
| {k13:99, k23:1} |
+-----+
```

1.6.6.6.23.3. MAP_KEYS

This topic describes the MAP_KEYS function and provides examples of using this function.

Syntax:

```
ARRAY map_keys(map<K, V>)
```

Description: This function returns all keys in the MAP parameter as an array.

Parameters: map: a value of the MAP type.

Return value: A value of the ARRAY type is returned. If the input value is NULL, NULL is returned.

Examples:

The t_table_map table contains two columns: c1 and t_map. c1 is of the BIGINT type and t_map is of the MAP<STRING,BIGINT> type.

```
+-----+-----+
| c1   | t_map |
+-----+-----+
| 1000 | {k11:86, k21:15} |
| 1001 | {k12:97, k22:2} |
| 1002 | {k13:99, k23:1} |
+-----+-----+
```

Execute the following statement:

```
select c1,map_keys(t_map) from t_table_map;
-- The following result is returned:
+-----+-----+
| c1   | _c1 |
+-----+-----+
| 1000 | [k11, k21] |
| 1001 | [k12, k22] |
| 1002 | [k13, k23] |
+-----+-----+
```

1.6.6.6.23.4. MAP_VALUES

This topic describes the MAP_VALUES function and provides examples of using this function.

Syntax:

```
ARRAY map_values(map<K, V>)
```

Description: This function returns all values in the map parameter as an array.

Parameters: map: a value of the MAP type.

Return value: A value of the ARRAY type is returned. If the input value is NULL, NULL is returned.

Examples:

```
select map_values(map('a',123,'b',456));
-- The following result is returned:
[123, 456]
```

1.6.6.6.23.5. SORT_ARRAY

This topic describes the SORT_ARRAY function and provides examples of using this function.

Syntax:

```
ARRAY sort_array(ARRAY<T>)
```

Description: This function sorts data in a given array.

Parameters: ARRAY<T>: data of the ARRAY type. Data in the array can be of any data type.

Return value: A value of the ARRAY type is returned.

Examples:

The t_array table contains three columns: c1, c2, and c3. The c1 and c3 columns are of the ARRAY<STRING> type, and the c2 column is of the ARRAY<INT> type.

```
+-----+-----+-----+
| c1   | c2   | c3   |
+-----+-----+-----+
| [a, c, f, b] | [4, 5, 7, 2, 5, 8] | [You, Me, Him] |
+-----+-----+-----+
```

Execute the following statement :

```
select sort_array(c1),sort_array(c2),sort_array(c3) from t_array;
-- The following result is returned:
[a, b, c, f] [2, 4, 5, 5, 7, 8] [Him, You, Me]
```

1.6.6.6.23.6. POSEXplode

This topic describes the POSEXplode function and provides examples of using this function.

Syntax:

```
posexplode(ARRAY<T>)
```

Description: This function converts a given array into a table that has two columns. The first column lists subscripts of each value in the array, starting from 0. The second column lists array elements.

Parameters: ARRAY<T>: a value of the ARRAY type. Data in the array can be of any data type.

Return value: The generated table is returned.

Examples:

```
select posexplode(array('a','c','f','b'));
-- The following result is returned:
+-----+-----+
| pos  | val |
+-----+-----+
| 0    | a   |
| 1    | c   |
| 2    | f   |
| 3    | b   |
+-----+-----+
```

1.6.6.6.23.7. STRUCT

This topic describes the STRUCT function and provides examples of using this function.

Syntax:

```
STRUCT struct(value1,value2, ...)
```

Description: This function creates a value of the STRUCT type based on the given value list.

Parameters: value: a value of any data type.

Return value: A value of the STRUCT type is returned. The fields in returned results are sequentially named as col1, col2,

Examples:

```
select struct('a',123,'ture',56.90);  
-- The following result is returned:  
{col1:a, col2:123, col3:ture, col4:56.9}
```

1.6.6.6.23.8. NAMED_STRUCT

This topic describes the NAMED_STRUCT function and provides examples of using this function.

Syntax:

```
STRUCT named_struct(string name1, T1 value1, string name2, T2 value2, ...)
```

Description: This function creates a value of the STRUCT type based on the given name/value list.

Parameters:

- value: a value of any data type.
- name: the name of the field of the STRING type. This parameter is a constant.

Return value: A value of the STRUCT type is returned. Fields in the return value are sequentially named as name1, name2,

Examples:

```
select named_struct('user_id',10001,'user_name','bob','married','F','weight',63.50);  
-- The following result is returned:  
{user_id:10001, user_name:bob, married:F, weight:63.5}
```

1.6.6.6.23.9. INLINE

This topic describes the INLINE function and provides examples of using this function.

Syntax:

```
inline(array<STRUCT<f1:T1, f2:T2, ... >>)
```

Description: This function expands a given STRUCT array. Each array element is given one row and each STRUCT element corresponds to one column in each row.

Parameters: STRUCT: the values in the array, which are of any data type.

Return value: The function generated by the table is returned.

Examples:

Column information of the t_table table is t_struct
 struct<user_id:bigint,user_name:string,married:string,weight:double>. Data in the t_table table:

```
+-----+
| t_struct |
+-----+
| {user_id:10001, user_name:LiLei, married:N, weight:63.5} |
| {user_id:10002, user_name:HanMeiMei, married:Y, weight:43.5} |
+-----+
```

Execute the following statement:

```
select inline(array(t_struct)) from t_table;
-- The following result is returned:
+-----+-----+-----+-----+
| user_id | user_name | married | weight |
+-----+-----+-----+-----+
| 10001   | LiLei     | N       | 63.5   |
| 10002   | HanMeiMei| Y       | 43.5   |
+-----+-----+-----+-----+
```

1.6.6.6.23.10. BETWEEN AND expression

This topic describes the BETWEEN AND expression function and provides examples of using this function.

Syntax:

```
A [NOT] BETWEEN B AND C
```

If the value of A, B, or C is NULL, NULL is returned. If the value of A is greater than or equal to the value of B and less than or equal to the value of C, True is returned. Otherwise, False is returned.

Examples:

The emp table contains the following data:

```
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
7369,SMITH,CLERK,7902,1980-12-17 00:00:00,800,,20
7499,ALLEN,SALESMAN,7698,1981-02-20 00:00:00,1600,300,30
7521,WARD,SALESMAN,7698,1981-02-22 00:00:00,1250,500,30
7566,JONES,MANAGER,7839,1981-04-02 00:00:00,2975,,20
7654,MARTIN,SALESMAN,7698,1981-09-28 00:00:00,1250,1400,30
7698,BLAKE,MANAGER,7839,1981-05-01 00:00:00,2850,,30
7782,CLARK,MANAGER,7839,1981-06-09 00:00:00,2450,,10
7788,SCOTT,ANALYST,7566,1987-04-19 00:00:00,3000,,20
7839,KING,PRESIDENT,,1981-11-17 00:00:00,5000,,10
7844,TURNER,SALESMAN,7698,1981-09-08 00:00:00,1500,0,30
7876,ADAMS,CLERK,7788,1987-05-23 00:00:00,1100,,20
7900,JAMES,CLERK,7698,1981-12-03 00:00:00,950,,30
7902,FORD,ANALYST,7566,1981-12-03 00:00:00,3000,,20
7934,MILLER,CLERK,7782,1982-01-23 00:00:00,1300,,10
7948,JACCKA,CLERK,7782,1981-04-12 00:00:00,5000,,10
7956,WELAN,CLERK,7649,1982-07-20 00:00:00,2450,,10
7956,TEBAGE,CLERK,7748,1982-12-30 00:00:00,1300,,10
```

Query the data with sal greater than or equal to 1000 and less than or equal to 1500.

```
select * from emp where sal BETWEEN 1000 and 1500;
-- The following result is returned:
+-----+-----+-----+-----+-----+-----+-----+-----+
| empno | ename | job | mgr | hiredate | sal | comm | deptno |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 00:00:00 | 1250.0 | 500.0 | 30 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 00:00:00 | 1250.0 | 1400.0 | 30 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 00:00:00 | 1500.0 | 0.0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1987-05-23 00:00:00 | 1100.0 | NULL | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 00:00:00 | 1300.0 | NULL | 10 |
| 7956 | TEBAGE | CLERK | 7748 | 1982-12-30 00:00:00 | 1300.0 | NULL | 10 |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

1.6.6.6.23.11. NVL

This topic describes the NVL function and provides an example of using this function.

Function declaration:

```
nvl(T value, T default_value)
```

Description: This function returns default_value if value is NULL. Otherwise, value is returned.

Example:

Assume that the t_data table has three columns: c1 of the STRING type, c2 of the BIGINT type, and c3 of the DATETIME type. The following data is contained in the table:

```
+---+-----+-----+
| c1 | c2 | c3 |
+---+-----+-----+
| NULL | 20 | 2017-11-13 05:00:00 |
| ddd | 25 | NULL |
| bbb | NULL | 2017-11-12 08:00:00 |
| aaa | 23 | 2017-11-11 00:00:00 |
+---+-----+-----+
```

After the NVL function is called, the NULL value in c1 is returned as 00000, the NULL value in c2 is returned as 0, and the NULL value in c3 is returned as a hyphen (-).

```
-- Execute the following statement:
select nvl(c1,'00000'),nvl(c2,0) nvl(c3,'-') from nv1_test;
-- Returned result:
+-----+-----+-----+
| _c0 | _c1 | _c2 |
+-----+-----+-----+
| bbb | 0 | 2017-11-12 08:00:00 |
| ddd | 25 | - |
| 00000 | 20 | 2017-11-13 05:00:00 |
| aaa | 23 | 2017-11-11 00:00:00 |
+-----+-----+-----+
```

1.6.6.6.23.12. TABLE_EXISTS

This topic describes the TABLE_EXISTS function and provides an example of using this function.

Function declaration:

```
boolean table_exists(string table_name)
```

Description: This function checks whether a specific table exists.

Parameters:

table_name: the table name of the STRING type. The value can include the project name, such as my_proj.my_table. If no project name is specified, the name of the current project is used.

Return value: A value of the BOOLEAN type is returned. If the specified table exists, True is returned. Otherwise, False is returned.

Example:

```
-- Use this function in a SELECT statement.
select if(table_exists('abd'), col1, col2) from src;
```

1.6.6.6.23.13. PARTITION_EXISTS

This topic describes the PARTITION_EXISTS function and provides an example of using this function.

Function declaration:

```
boolean partition_exists(string table_name, string... partitions)
```

Description: This function checks whether a specific partition exists.

Parameters:

- table_name: the table name of the STRING type. The value can include the project name, such as my_proj.my_table. If no project name is specified, the name of the current project is used.
- partitions: the partition names of the STRING type. In this parameter, you must specify the values of partition key columns in a table based on the sequence of these columns. The number of values must be the same as the number of partition key columns.

Return value: A value of the BOOLEAN type is returned. If the specified partitions exist, True is returned. Otherwise, False is returned.

Example:

```
create table foo (id bigint) partitioned by (ds string, hr string);
-- Create a partitioned table named foo.
alter table foo add partition (ds='20190101', hr='1');
-- Add a partition to foo.
select partition_exists('foo', '20190101', '1');
-- Check whether partitions with ds='20190101' and hr='1' exist.
```

1.6.7. UDFs

1.6.7.1. Overview

This topic describes user-defined functions (UDFs) in MaxCompute.

MaxCompute offers a variety of built-in functions to meet your computing requirements. You can also use UDFs.

UDFs work in a way similar to built-in functions. For the mapping between Java and MaxCompute data types, see [Types of parameters and returned values](#).

Note If a UDF has the same name as a built-in function, MaxCompute executes the UDF by default. To call the built-in function, you must execute the `select ::Function name (expression);` statement. For example, if a UDF in MaxCompute is named CONCAT, MaxCompute calls the UDF rather than the CONCAT built-in function. To call the CONCAT built-in function, execute the `select ::concat('ab', 'c');` statement.

The following table describes three types of UDFs supported by MaxCompute.

UDF type	Description
UDF	User-defined scalar function. The input and output of a user-defined scalar function have a one-to-one mapping relationship. Each time a user-defined scalar function reads one row of data, one value is returned.
UDTF	User-defined table-valued function. A UDTF is used in scenarios in which multiple rows of data are returned after you call the function. Only this type of function can return multiple fields. A UDTF is not equivalent to a user-defined type (UDT).
UDAF	User-defined aggregate function. The input and output of a user-defined aggregate function have a many-to-one mapping relationship. Multiple input records are aggregated to generate one output value. This type of function can be used with the GROUP BY clause of SQL.

Note When you use a UDF in an SQL statement, the system may prompt insufficient memory. The reason is that the memory size required for the computing task exceeds the default memory size. In this case, you can execute the `set odps.sql.udf.joiner.jvm.memory=xxxx;` statement to adjust the memory size.

MaxCompute UDFs can be shared by multiple projects. UDFs in one project can be used in another project. SQL statement for cross-project UDF sharing:

```
select other_project:udf_in_other_project(arg0, arg1) as res from table_t;
```

If you use Maven, you can obtain it from the [Maven repository](#).

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>odps-sdk-udf</artifactId>
  <version>0.29.10-public</version>
</dependency>
```

1.6.7.2. Types of parameters and return values

This topic describes the types of parameters and return values of user-defined functions (UDFs) in MaxCompute.

Java UDFs in MaxCompute V2.0 support more basic data types in addition to BIGINT, STRING, DOUBLE, and BOOLEAN. These UDFs also support complex data types, such as ARRAY, MAP, and STRUCT, as well as Writable types.

To use a new basic data type, UDFs specify their function signatures in the following ways:

- A user-defined aggregate function (UDAF) or user-defined table-valued function (UDTF) uses the @Resolve annotation, for example, @Resolve("smallint->varchar(10)").
- A user-defined scalar function uses the evaluate method. In this case, the built-in function types of MaxCompute and Java function types have a one-to-one mapping relationship.

 Notice

- You can use type,* to pass any number of parameters, for example, @resolve("string,*->array<string>"). You must add Subtype after ARRAY.
- The field name and field type of com.aliyun.odps.data.Struct cannot be reflected. Therefore, you must use the @Resolve annotation to obtain the field name and field type. In other words, to use STRUCT in a UDF, you must add the @Resolve annotation to the UDF class. This annotation affects only the overloads of parameters or return values that contain com.aliyun.odps.data.Struct.
- Only one @Resolve annotation can be provided for the class. Therefore, only one overload with a STRUCT parameter or return value exists in a UDF.

To use a complex data type, UDFs specify their function signatures in the following ways:

- A UDTF uses the @Resolve annotation, for example, @Resolve("array<string>,struct<a1:bigint,b1:string>,string->map<string,bigint>,struct<b1:bigint>").
- A user-defined scalar function uses the signature of the evaluate method to match input and output types. This involves the following mapping between MaxCompute and Java data types:
 - ARRAY in MaxCompute maps to java.util.List.
 - MAP in MaxCompute maps to java.util.Map.
 - STRUCT in MaxCompute maps to com.aliyun.odps.data.Struct.
- A UDAF uses the @Resolve annotation, for example, @Resolve("smallint->varchar(10)").

The following table describes the mapping between MaxCompute and Java data types.

MaxCompute data type	Java data type
TINYINT	java.lang.Byte
SMALLINT	java.lang.Short
INT	java.lang.Integer
BIGINT	java.lang.Long
FLOAT	java.lang.Float
DOUBLE	java.lang.Double
DECIMAL	java.math.BigDecimal
BOOLEAN	java.lang.Boolean
STRING	java.lang.String
VARCHAR	com.aliyun.odps.data.Varchar
BINARY	com.aliyun.odps.data.Binary

MaxCompute data type	Java data type
DATETIME	java.util.Date
TIMESTAMP	java.sql.Timestamp
ARRAY	java.util.List
MAP	java.util.Map
STRUCT	com.aliyun.odps.data.Struct

 **Note**

- Make sure that the input and output parameters in a UDF are of a Java type. Otherwise, error ODPS-0130071 is returned.
- Java data types and the data types of return values are objects. Java data types must start with an uppercase letter.
- The NULL value in SQL statements is represented by a NULL reference in Java. Java Primitive Type cannot have NULL values and must not be used.
- The ARRAY type in the preceding table maps java.util.List.

MaxCompute V2.0 allows you to use Writable types as parameters and return values when you define Java UDFs. The following table describes the mapping between MaxCompute data types and Java Writable types.

MaxCompute data type	Java Writable type
TINYINT	ByteWritable
SMALLINT	ShortWritable
INT	IntWritable
BIGINT	LongWritable
FLOAT	FloatWritable
DOUBLE	DoubleWritable
DECIMAL	BigDecimalWritable
BOOLEAN	BooleanWritable
STRING	Text
VARCHAR	VarcharWritable
BINARY	BytesWritable
DATETIME	DatetimeWritable
TIMESTAMP	TimestampWritable
INTERVAL_YEAR_MONTH	IntervalYearMonthWritable

MaxCompute data type	Java Writable type
INTERVAL_DAY_TIME	IntervalDayTimeWritable
ARRAY	N/A
MAP	N/A
STRUCT	N/A

1.6.7.3. UDF

This topic describes user-defined scalar functions (UDFs) and provides examples of using UDFs.

UDFs must inherit the class `com.aliyun.odps.udf.UDF` and implement the `evaluate` method of the class. The `evaluate` method must be a non-static public method, and the types of parameters and return values are used as the UDF signature in SQL statements. You can implement multiple `evaluate` methods in a UDF. To call a UDF, the framework matches the correct `evaluate` method based on the parameter type called by the UDF.

Note

- We recommend that you do not use the classes that have the same name but different function logic in different JAR packages. For example, in UDF(UDAF/UDTF): `udf1`, `udf2`, the JAR package of `udf1` is `udf1.jar` and the JAR package of `udf2` is `udf2.jar`. Assume that the two JAR packages contain the `com.aliyun.UserFunction.class` class. If you use the two UDFs in the same SQL statement, the system randomly loads the class contained in one of the two JAR packages. This may cause the UDFs to be executed in different ways or even cause a compilation failure.
- If you execute an SQL statement that has two UDFs, the UDFs are not isolated from each other. This is because the UDFs share the same classpath. If the resources referenced by the UDFs contain the same class, the class that the classloader attempts to load is uncertain. To avoid this issue, make sure that the resources referenced by the two UDFs do not contain the same class.

You can use `void setup(ExecutionContext ctx)` to initialize a UDF and use `void close()` to terminate a UDF.

The following example shows how to implement a UDF:

```
package org.alidata.odps.udf.examples;
import com.aliyun.odps.udf.UDF;
public final class Lower extends UDF {
    public String evaluate(String s) {
        if (s == null) {
            return null;
        }
        return s.toLowerCase();
    }
}
```

The following example shows how to implement `Concat` by using a Writable type:

```
package com.aliyun.odps.udf.example;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.udf.UDF;
public class MyConcat extends UDF {
    private Text ret = new Text();
    public Text evaluate(Text a, Text b) {
        if (a == null || b == null) {
            return null;
        }
        ret.clear();
        ret.append(a.getBytes(), 0, a.getLength());
        ret.append(b.getBytes(), 0, b.getLength());
        return ret;
    }
}
```

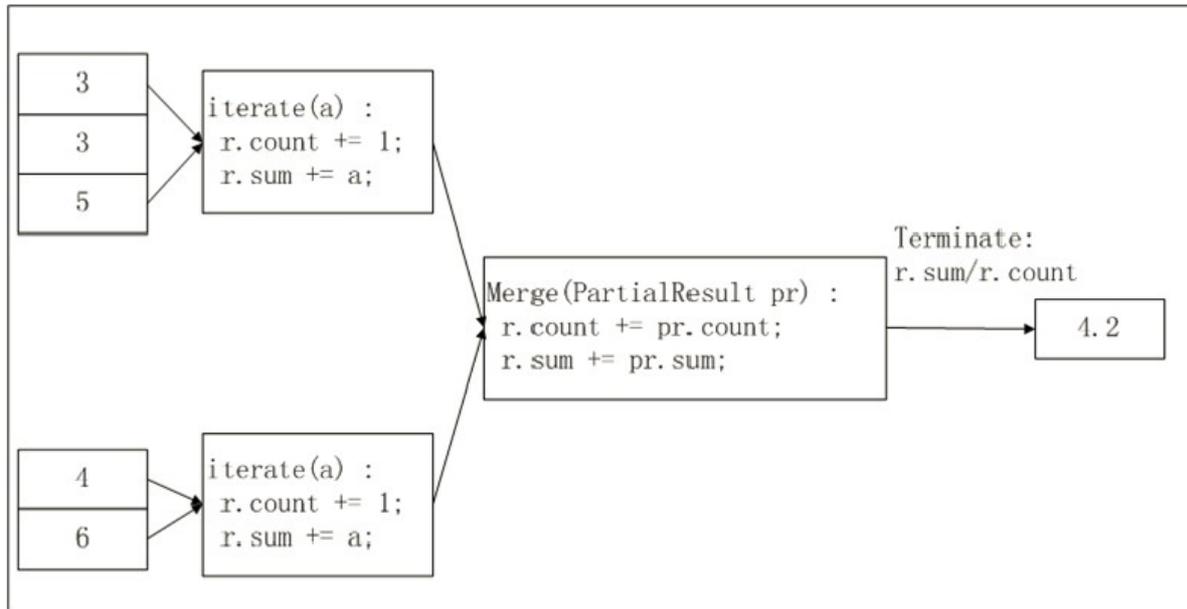
1.6.7.4. UDAF

This topic describes user-defined aggregate functions (UDAFs) and provides examples of using UDAFs.

UDAFs must inherit the `com.aliyun.odps.udf.Aggregator` class and implement the following methods:

```
import com.aliyun.odps.udf.ContextFunction;
import com.aliyun.odps.udf.ExecutionContext;
import com.aliyun.odps.udf.UDFException;
public abstract class Aggregator implements ContextFunction {
    @Override
    public void setup(ExecutionContext ctx) throws UDFException {
    }
    @Override
    public void close() throws UDFException {
    }
    /**
     * Create an aggregation buffer * @return Writable aggregation buffer
     */
    abstract public Writable newBuffer();
    /**
     * @param buffer: aggregation buffer * @param args: a parameter specified to call a UDAF in SQL. It cannot be NULL, but the values in args can be NULL, which indicates that the input data is NULL * @throws UDFException.
     */
    abstract public void iterate(Writable buffer, Writable[] args) throws UDFException;
    /**
     * Generate the final result * @param buffer * @return Final result of Object UDAF * @throws UDFException.
     */
    abstract public Writable terminate(Writable buffer) throws UDFException;
    abstract public void merge(Writable buffer, Writable partial) throws UDFException;
}
```

The most important methods are `iterate`, `merge`, and `terminate` because the main logic of UDAFs relies on these methods. In addition, you must implement the user-defined Writable buffer. The following figure illustrates the implementation logic and procedure to calculate the avg value by using a MaxCompute UDAF.



In the preceding figure, the input data is sliced based on the specified size. The size of each slice is suitable for a worker to complete the calculation in a specified time. The slice size must be manually configured.

The calculation procedure of a UDAF involves two steps:

- Step 1: Each worker counts the data quantity and total sum in a slice. You can consider the data quantity and total sum in each slice as an intermediate result.
- Step 2: Each worker gathers the information about each slice generated in Step 1. In the final output, `r.sum/r.count` is the average value of all input data.

Sample code for a UDAF that is used to calculate the average value:

```
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import com.aliyun.odps.io.DoubleWritable;
import com.aliyun.odps.io.Writable;
import com.aliyun.odps.udf.Aggregator;
import com.aliyun.odps.udf.UDFException;
import com.aliyun.odps.udf.annotation.Resolve;
@Resolve("double->double")
public class AggrAvg extends Aggregator {
    private static class AvgBuffer implements Writable {
        private double sum = 0;
        private long count = 0;
        @Override
        public void write(DataOutput out) throws IOException {
            out.writeDouble(sum);
            out.writeLong(count);
        }
        @Override
        public void readFields(DataInput in) throws IOException {
            sum = in.readDouble();
            count = in.readLong();
        }
    }
    private DoubleWritable ret = new DoubleWritable();
    @Override
    public Writable newBuffer() {
        return new AvgBuffer();
    }
    @Override
    public void iterate(Writable buffer, Writable[] args) throws UDFException {
        DoubleWritable arg = (DoubleWritable) args[0];
        AvgBuffer buf = (AvgBuffer) buffer;
        if (arg != null) {
            buf.count += 1;
            buf.sum += arg.get();
        }
    }
    @Override
    public Writable terminate(Writable buffer) throws UDFException {
        AvgBuffer buf = (AvgBuffer) buffer;
        if (buf.count == 0) {
            ret.set(0);
        } else {
            ret.set(buf.sum / buf.count);
        }
        return ret;
    }
    @Override
    public void merge(Writable buffer, Writable partial) throws UDFException {
        AvgBuffer buf = (AvgBuffer) buffer;
        AvgBuffer p = (AvgBuffer) partial;
        buf.sum += p.sum;
        buf.count += p.count;
    }
}
```

Sample code for implementing Concat by using Writable types:

```
package com.aliyun.odps.udf.example;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.udf.UDF;
public class MyConcat extends UDF {
    private Text ret = new Text();
    public Text evaluate(Text a, Text b) {
        if (a == null || b == null) {
            return null;
        }
        ret.clear();
        ret.append(a.getBytes(), 0, a.getLength());
        ret.append(b.getBytes(), 0, b.getLength());
        return ret;
    }
}
```

1.6.7.5. UDTFs

1.6.7.5.1. Overview

This topic describes user-defined table-valued functions (UDTFs) and provides examples of using UDTFs.

UDTFs must inherit the `com.aliyun.odps.udf.UDTF` class and implement four methods. The following table describes the methods.

Method	Description
<code>public void setup(ExecutionContext ctx) throws UDFException</code>	The initialization method to call the user-defined initialization behavior before a UDTF processes the input data. <code>setup</code> is called once first for each worker.
<code>public void process(Object[] args) throws UDFException</code>	This method is called by the framework. Each SQL record calls <code>process</code> once. The parameters of <code>process</code> are the input parameters of the UDTF specified in the SQL statement. The input parameters are passed in as <code>Object[]</code> , and the results are returned by using the <code>forward</code> function. You must call <code>forward</code> in the <code>process</code> function to determine the output data.
<code>public void close() throws UDFException</code>	The method for terminating a UDTF. The framework calls this method only once. The method is called after the last record is processed.
<code>public void forward(Object ...o) throws UDFException</code>	You can call the <code>forward</code> method to return data. One record is returned each time <code>forward</code> is called. The record corresponds to the column specified by the AS clause in the SQL statement of the UDTF.

Example:

```
package org.alidata.odps.udtf.examples;
import com.aliyun.odps.udf.UDTF;
import com.aliyun.odps.udf.UDTFCollector;
import com.aliyun.odps.udf.annotation.Resolve;
import com.aliyun.odps.udf.UDFException;
// TODO define input and output types, e.g., "string,string->string,bigint".
@Resolve("string,bigint->string,bigint")
public class MyUDTF extends UDTF {
    @Override
    public void process(Object[] args) throws UDFException {
        String a = (String) args[0];
        Long b = (Long) args[1];
        for (String t: a.split("\\s+")) {
            forward(t, b);
        }
    }
}
```

Assume that you create a UDTF in MaxCompute with the registered function name of `user_udtf`. Execute the following statement to use the UDTF:

```
select user_udtf(col0, col1) as (c0, c1) from my_table;
```

The following example shows the values of `col0` and `col1` in `my_table`.

```
+-----+-----+
| col0 | col1 |
+-----+-----+
| A B | 1   |
| C D | 2   |
+-----+-----+
```

The following example shows the execution result of the `SELECT` statement.

```
+----+----+
| c0 | c1 |
+----+----+
| A | 1 |
| B | 1 |
| C | 2 |
| D | 2 |
+----+----+
```

1.6.7.5.2. Examples of using UDTFs

This topic provides examples of using UDTFs.

Use a UDTF to read resources in MaxCompute

You can use a UDTF to read resources in MaxCompute. Example:

1. Compile a UDTF program. After the compilation is successful, export the JAR package. In this example, the JAR package is named `udtfexample1.jar`.

```
package com.aliyun.odps.examples.udf;
```

```

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.Iterator;
import com.aliyun.odps.udf.ExecutionContext;
import com.aliyun.odps.udf.UDFException;
import com.aliyun.odps.udf.UDTF;
import com.aliyun.odps.udf.annotation.Resolve;
/**
 * project: example_project
 * table: wc_in2
 * partitions: p2=1,p1=2
 * columns: colc,colb
 */
@Resolve("string,string->string,bigint,string")
public class UDTFResource extends UDTF {
    ExecutionContext ctx;
    long fileResourceLineCount;
    long tableResource1RecordCount;
    long tableResource2RecordCount;
    @Override
    public void setup(ExecutionContext ctx) throws UDFException {
        this.ctx = ctx;
        try {
            InputStream in = ctx.readResourceFileAsStream("file_resource.txt");
            BufferedReader br = new BufferedReader(new InputStreamReader(in));
            String line;
            fileResourceLineCount = 0;
            while ((line = br.readLine()) != null) {
                fileResourceLineCount++;
            }
            br.close();
            Iterator<Object[]> iterator = ctx.readResourceTable("table_resource1").iterator();
            tableResource1RecordCount = 0;
            while (iterator.hasNext()) {
                tableResource1RecordCount++;
                iterator.next();
            }
            iterator = ctx.readResourceTable("table_resource2").iterator();
            tableResource2RecordCount = 0;
            while (iterator.hasNext()) {
                tableResource2RecordCount++;
                iterator.next();
            }
        } catch (IOException e) {
            throw new UDFException(e);
        }
    }
    @Override
    public void process(Object[] args) throws UDFException {
        String a = (String) args[0];
        long b = args[1] == null ? 0 : ((String) args[1]).length();
        forward(a, b, "fileResourceLineCount=" + fileResourceLineCount + "|tableResource1RecordCount="
            + tableResource1RecordCount + "|tableResource2RecordCount=" + tableResource2RecordCount);
    }
}

```

2. Add resources to MaxCompute.

```
Add file file_resource.txt;
Add jar udtfexample1.jar;
Add table table_resource1 as table_resource1;
Add table table_resource2 as table_resource2;
```

3. Create the my_udtf UDTF in MaxCompute.

```
create function mp_udtf as com.aliyun.odps.examples.udf.UDTFResource using
'udtfexample1.jar, file_resource.txt, table_resource1, table_resource2';
```

4. Run this UDTF.

```
select mp_udtf("10", "20") as (a, b, fileResourceLineCount) from table_resource1;
```

Returned result:

```
+-----+-----+-----+
| a | b | fileResourceLineCount |
+-----+-----+-----+
| 10 | 2 | fileResourceLineCount=3|tableResource1RecordCount=0|tableResource2RecordCount=0 |
| 10 | 2 | fileResourceLineCount=3|tableResource1RecordCount=0|tableResource2RecordCount=0 |
+-----+-----+-----+
```

Use complex data types in a UDF

The following example defines a UDF with three overloads. The first overload uses the ARRAY type, the second overload uses the MAP type, and the third overload uses the STRUCT type. The third overload uses STRUCT as the data type of parameters or returned values. Therefore, the @Resolve annotation must be added to the UDF class to specify the STRUCT type.

```
import com.aliyun.odps.udf.UDF;
import com.aliyun.odps.udf.annotation.Resolve;
@Resolve("struct,string->string")
public class UdfArray extends UDF {
    public String evaluate(List vals, Long len) {
        return vals.get(len.intValue());
    }
    public String evaluate(Map map, String key) {
        return map.get(key);
    }
    public String evaluate(Struct struct, String key) {
        return struct.getFieldValue("a") + key;
    }
}
```

You can pass complex data types into a UDF.

```
create function my_index as 'UdfArray' using 'myjar.jar';
select id, my_index(array('red', 'yellow', 'green'), colorOrdinal) as color_name from co
```

Use Hive UDFs

MaxCompute V2.0 supports Hive UDFs. Some Hive UDFs and UDTFs can be directly used in MaxCompute.

 **Note** MaxCompute V2.0 is compatible with Hive 2.1.0 and Hadoop 2.7.2. If you develop a UDF by using other versions of Hive or Hadoop, you may need to recompile the UDF on the compatible Hive or Hadoop version.

The following example shows how to use a Hive UDF in MaxCompute:

```
package com.aliyun.odps.compiler.hive;
import org.apache.hadoop.hive.ql.exec.UDFArgumentException;
import org.apache.hadoop.hive.ql.metadata.HiveException;
import org.apache.hadoop.hive.ql.udf.generic.GenericUDF;
import org.apache.hadoop.hive.serde2.objectinspector.ObjectInspector;
import org.apache.hadoop.hive.serde2.objectinspector.ObjectInspectorFactory;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;
public class Collect extends GenericUDF {
    @Override
    public ObjectInspector initialize(ObjectInspector[] objectInspectors) throws UDFArgumentException {
        if (objectInspectors.length == 0) {
            throw new UDFArgumentException("Collect: input args should >= 1");
        }
        for (int i = 1; i < objectInspectors.length; i++) {
            if (objectInspectors[i] != objectInspectors[0]) {
                throw new UDFArgumentException("Collect: input oi should be the same for all args");
            }
        }
        return ObjectInspectorFactory.getStandardListObjectInspector(objectInspectors[0]);
    }
    @Override
    public Object evaluate(DeferredObject[] deferredObjects) throws HiveException {
        List<Object> objectList = new ArrayList<>(deferredObjects.length);
        for (DeferredObject deferredObject : deferredObjects) {
            objectList.add(deferredObject.get());
        }
        return objectList;
    }
    @Override
    public String getDisplayString(String[] strings) {
        return "Collect";
    }
}
```

The Hive UDF in the example packages any type and number of parameters into arrays. In this example, the output JAR package is named test.jar.

```
-- Add a resource.
Add jar test.jar;
-- Create a Hive UDF.
CREATE FUNCTION hive_collect as 'com.aliyun.odps.compiler.hive.Collect' using 'test.jar';
-- Use the Hive UDF.
set odps.sql.hive.compatible=true;
select hive_collect(4y,5y,6y) from dual;
```

Returned result :

```
+-----+
|_c0 |
+-----+
|[4, 5, 6]|
+-----+
```

 **Note** Take note of the following items when you use Hive UDFs that are compatible with MaxCompute:

- Specify the JAR package when you create a Hive UDF. MaxCompute cannot automatically add all JAR packages to the classpath. The `add jar` command in MaxCompute creates a permanent resource in the project.
- Insert `set odps.sql.hive.compatible=true;` before an SQL statement, and commit and execute the SET statement with the SQL statement.
- Pay attention to the limits imposed by the Java sandbox on MaxCompute.

1.6.7.6. Python UDFs

1.6.7.6.1. Limits

This topic describes limits on Python user-defined functions (UDFs).

MaxCompute uses Python 2.7. User code runs in a sandbox, which is a limited environment. In this environment, the following operations are prohibited:

- Read and write local files.
- Start subprocesses.
- Start threads.
- Initiate socket-based communication.
- Call other systems.

Due to these limits, the code that you upload must be implemented by standard Python, and C extension modules are not allowed in your code.

In addition, the modules that involve the preceding operations are disabled in the Python standard library. The following modules are available in the standard library:

- The modules that are implemented by standard Python.
- The following C extension modules:
 - array
 - audioop
 - binascii
 - _bisect
 - cmath
 - _codecs_cn
 - _codecs_hk
 - _codecs_iso2022
 - _codecs_jp
 - _codecs_kr
 - _codecs_tw

- `_collections`
 - `cStringIO`
 - `datetime`
 - `_functools`
 - `future_builtins`
 - `_hashlib`
 - `_heapq`
 - `itertools`
 - `_json`
 - `_locale`
 - `_lsprof`
 - `math`
 - `_md5`
 - `_multibytecodec`
 - `operator`
 - `_random`
 - `_sha256`
 - `_sha512`
 - `_sha`
 - `_struct`
 - `strop`
 - `time`
 - `unicodedata`
 - `_weakref`
 - `cPickle`
- Some modules with limited functionality. For example, the size of data that your code can write into standard output `sys.stdout` and standard error output `sys.stderr` in a sandbox is limited to 20 KB. Characters that exceed this limit are ignored.

1.6.7.6.2. Third-party libraries

This topic describes third-party libraries for Python user-defined functions (UDFs).

Common third-party libraries such as NumPy are installed in the runtime environment to supplement the standard library.

 **Warning** The use of third-party libraries is also subject to limits. For example, when you use a third-party library, you are not allowed to access local data and you can use only limited network I/O resources. The related APIs in the third-party libraries are disabled.

1.6.7.6.3. Types of parameters and return values

This topic describes the types of parameters and return values of Python user-defined functions (UDFs).

The types of parameters and return values are specified by using the following method:

```
@odps.udf.annotate(signature)
```

Python UDFs support MaxCompute SQL data types, such as BIGINT, STRING, DOUBLE, BOOLEAN, DATETIME, DECIMAL, complex data types, and nested complex data types. Complex data types include ARRAY, MAP, and STRUCT. Before you execute an SQL statement, the types of parameters and return values of all functions must be determined. Python is a dynamically typed language. You must add decorators to the UDF class to specify the function signature.

The function signature is specified by a string. The following syntax is supported:

```
arg_type_list '->' type_list
arg_type_list: type_list | '*' | '' | 'char(n)' | 'varchar(n)'
type_list: [type_list ',' ] type
type: 'bigint' | 'string' | 'double' | 'boolean' | 'datetime' | 'float' | 'binary' | 'date' | 'decimal' | 'decimal(precision,scale)'
```

 Note

- The part to the left of the arrow indicates the types of parameters. The part to the right of the arrow indicates the types of return values.
- The return value of a UDTF can contain multiple columns, while the return value of a UDF or UDAF contains only one column.
- An asterisk (*) represents variable-length parameters. If variable-length parameters are used, UDFs, UDTFs, and UDAFs can match input parameters of any type.
- In a project that uses the MaxCompute V2.0 data type edition, you can set the decimal parameter to precision or scale.
- The return value cannot be of the CHAR or VARCHAR type.

The following example shows a valid signature:

```
'bigint,double->string' # Parameters are of the BIGINT and DOUBLE types, and return values are of the STRING type.
'bigint,boolean->string,datetime' # UDTF parameters are of the BIGINT and BOOLEAN types, and return values are of the STRING and DATETIME types.
'*->string' # Input parameters are variable-length parameters and can be of any type, and return values are of the STRING type.
'->double' # Parameters are left empty, and return values are of the DOUBLE type.
'array<bigint>->struct<x:string, y:int>' # Parameters are of the ARRAY<BIGINT> type, and return values are of the STRUCT<x:STRING, y:BIGINT> type.
'>map<bigint, string>' # Parameters are left empty, and return values are of the MAP<BIGINT, STRING> type.
```

If an invalid signature is found during query parsing, an error is reported, and the execution of the function is prohibited. During the execution of the function, the UDF parameters of the type specified by the function signature are passed. The return values must be of the same type as those specified by the function signature. Otherwise, an error is reported. The following table describes the mappings between Python data types and MaxCompute SQL data types.

MaxCompute SQL data type	Python data type
BIGINT	INT
STRING	STR
DOUBLE	FLOAT
BOOLEAN	BOOL

MaxCompute SQL data type	Python data type
DATETIME	INT
FLOAT	FLOAT
CHAR	STR
VARCHAR	STR
BINARY	BYTEARRAY
DATE	INT
DECIMAL	DECIMAL.DECIMAL
ARRAY	LIST
MAP	DICT
STRUCT	COLLECTIONS.NAMEDTUPLE

 **Note**

- A value of the DATETIME type is converted to a value of the INT type. The value of the INT type follows the UNIX format, which is the number of milliseconds that have elapsed since 00:00:00 Thursday, January 1, 1970. You can process data of the DATETIME type by using the DATETIME module in the Python standard library.
- silent is added to `odps.udf.int(value,[silent=True])`. If silent is set to True and value cannot be converted to the INT type, None is returned, and no error is reported.
- The NULL value corresponds to None in Python.

1.6.7.6.4. UDF

This topic describes user-defined scalar functions (UDFs) and provides an example of using UDFs.

To implement Python UDFs, you must define a new-style class and implement the evaluate method.

```
from odps.udf import annotate
@annotate("bigint,bigint->bigint")
class MyPlus(object):
    def evaluate(self, arg0, arg1):
        if None in (arg0, arg1):
            return None
        return arg0 + arg1
```

 **Note** A Python UDF must have its signature specified by using `annotate`.

1.6.7.6.5. UDAF

This topic describes user-defined aggregate functions (UDAFs) and provides an example of using UDAFs.

- class `odps.udf.BaseUDAF`: is inherited to implement Python UDAFs.

- `BaseUDAF.new_buffer()`: returns buffer of the intermediate value of a UDAF. buffer must be a marshallable object (such as LIST or DICT), and the buffer size cannot increase with the data volume. Under extreme circumstances, the buffer size cannot exceed 2 MB after the marshal operation.
- `BaseUDAF.iterate(buffer[, args, ...])`: aggregates args to buffer of the intermediate value.
- `BaseUDAF.merge(buffer, pBuffer)`: aggregates buffer of two intermediate values. It merges pBuffer to buffer.
- `BaseUDAF.terminate(buffer)`: converts buffer of the intermediate value to a value of a basic data type in MaxCompute SQL.

Example of using a UDAF to obtain the average value:

```
@annotate('double->double')
class Average(BaseUDAF):
    def new_buffer(self):
        return [0, 0]
    def iterate(self, buffer, number):
        if number is not None:
            buffer[0] += number
            buffer[1] += 1
    def merge(self, buffer, pBuffer):
        buffer[0] += pBuffer[0]
        buffer[1] += pBuffer[1]
    def terminate(self, buffer):
        if buffer[1] == 0:
            return 0.0
        return buffer[0] / buffer[1]
```

1.6.7.6.6. UDTF

This topic describes user-defined table-valued functions (UDTFs) and provides an example of using UDTFs.

- `class odps.udf.BaseUDTF`: the base class of Python UDTFs. This class can be inherited to implement the `PROCESS` and `CLOSE` methods.
- `BaseUDTF.__init__()`: the initialization method. To implement this method for a derived class, you must call the `super(BaseUDTF, self).__init__()` initialization method of the base class at the beginning of code execution. Throughout the lifecycle of a UDTF, the `INIT` method is called only once. It is called only before the first record is processed. If a UDTF needs to save internal states, all states can be initialized by using this method.
- `BaseUDTF.process([args, ...])`: is called by the MaxCompute SQL framework. The `PROCESS` method is called for each record in SQL. The parameters in the `PROCESS` method are the UDTF input parameters that are specified in SQL statements.
- `BaseUDTF.forward([args, ...])`: the UDTF output method. This method is called by user code. One record is generated each time the `FORWARD` method is called. The parameters in the `FORWARD` method are the UDTF output parameters that are specified in SQL statements.
- `BaseUDTF.close()`: the UDTF termination method. This method is called only once by the MaxCompute SQL framework. It is called only after the last record is processed.

Example:

```
#coding:utf-8
# explode.py
from odps.udf import annotate
from odps.udf import BaseUDTF
@annotate('string -> string')
class Explode(BaseUDTF):
    # Use commas (,) to separate the string into multiple records.
    def process(self, arg):
        props = arg.split(',')
        for p in props:
            self.forward(p)
```

 **Note** The types of parameters and the return value of a Python UDTF can be specified without the need to use `annotate` to specify the UDTF signature. This way, the function can match input parameters of any type in SQL. However, the type of the return value cannot be deduced. All output parameters are considered to be of the `STRING` type. Therefore, when `FORWARD` is called, all output values must be converted to the `STRING` type.

1.6.7.6.7. Resource reference

This topic describes how to reference resources in Python user-defined functions (UDFs) and provides an example for reference.

You can reference file and table resources in Python UDFs by using the `odps.distcache` module.

The following sample code shows the syntax that is used to reference file resources:

```
odps.distcache.get_cache_file(resource_name)
-- Return the content of a specified resource.
```

The `resource_name` parameter is a string that corresponds to the name of an existing file resource in the current project. If the resource name is invalid or the file resource does not exist, an error is returned.

The return value is a file-like object. If this object is no longer used, the caller must call the `close` method to release the open file resource.

Example:

```
@annotate('bigint->string')
class DistCacheExample(object):
    def __init__(self):
        cache_file = get_cache_file('test_distcache.txt')
        kv = {}
        for line in cache_file:
            line = line.strip()
            if not line:
                continue
            k, v = line.split()
            kv[int(k)] = v
        cache_file.close()
        self.kv = kv
    def evaluate(self, arg):
        return self.kv.get(arg)
```

The following sample code shows the syntax that is used to reference table resources:

```
odps.distcache.get_cache_table(resource_name)
```

The `resource_name` parameter is a string that corresponds to the name of an existing table resource in the current project. If the resource name is invalid or the table resource does not exist, an error is returned.

The return value is of the GENERATOR type. The caller traverses the table to obtain the content. A record of the ARRAY type is obtained each time the caller traverses the table.

Example:

```
from odps.udf import annotate
from odps.distcache import get_cache_table
@annotate('->string')
class DistCacheTableExample(object):
    def __init__(self):
        self.records = list(get_cache_table('udf_test'))
        self.counter = 0
        self.ln = len(self.records)
    def evaluate(self):
        if self.counter > self.ln - 1:
            return None
        ret = self.records[self.counter]
        self.counter += 1
        return str(ret)
```

1.6.7.7. Python 3 UDF

This topic describes the precautions when you use Python 3 user-defined functions (UDFs).

Python Software Foundation announces the End of Life (EOL) for Python 2. Due to this reason, MaxCompute supports Python 3 and uses Python 3.7.3.

Limits

Python 3 is not compatible with Python 2. You can select Python 2 or Python 3 to execute an SQL statement. However, you cannot use Python 2 code and Python 3 code in a single SQL statement at the same time.

Enable Python 3

You can enable Python 3 only for jobs. To enable Python 3 for a job, add the following command before the required SQL statement and execute them together:

```
set odps.sql.python.version=cp37;
```

Use third-party libraries

MaxCompute Python UDFs support third-party libraries. The third-party library NumPy is installed in the Python 2 runtime environment to supplement the standard library.

NumPy is not installed in the Python 3 runtime environment in MaxCompute. To use a NumPy UDF, you must manually upload a NumPy wheel package. If you download a NumPy wheel package from [Python Package Index \(PyPI\)](#), the package name is `numpy-<Version>-cp37-cp37m-manylinux1_x86_64.whl`.

Port Python 2 UDFs

Python Software Foundation announces the EOL for Python 2. Therefore, we recommend that you port Python 2 UDFs. The method used to port Python 2 UDFs varies based on projects.

- In a new project or an existing project where no Python UDFs exist, we recommend that you use Python 3 to write all Python UDFs.
- Exercise caution when you enable Python 3 in an existing project where a large number of Python 2 UDFs exist. This is because you cannot use Python 2 code and Python 3 code in a single SQL statement at the same time. If you plan to gradually replace Python 2 UDFs with Python 3 UDFs, use the following methods:
 - Use Python 3 to write new UDFs and enable Python 3 for new jobs.
 - Rewrite Python 2 UDFs to make them compatible with both Python 2 and Python 3. For more information about how to rewrite UDFs, see [Porting Python 2 code to Python 3](#).

 **Note** If you write a public UDF that needs to be shared among multiple projects, we recommend that you write the UDF to be compatible with both Python 2 and Python 3.

1.6.7.8. SQL Function

This topic describes how to use SQL functions. SQL functions allow you to reference SQL user-defined functions (UDFs) in SQL scripts.

Background information

You can use SQL functions of MaxCompute to resolve the following issues:

- Generally, a large amount of similar code exists, which is inconvenient to maintain and prone to errors. To use UDFs, you must develop code, compile the code in Java, and then create resources and functions. The process is complicated. In addition, UDFs cannot catch up with built-in functions in terms of performance. Example:

```
SELECT
  NVL(STR_TO_MAP(GET_JSON_OBJECT(col, '$.key1')), 'default') AS key1,
  NVL(STR_TO_MAP(GET_JSON_OBJECT(col, '$.key2')), 'default') AS key2,
  ...
  NVL(STR_TO_MAP(GET_JSON_OBJECT(col, '$.keyN')), 'default') AS keyN
FROM t;
```

- One function can be transferred as a parameter to another only by using code similar to lambda expressions in Java.

Feature description

As a type of UDFs, SQL functions allow you to create UDFs in SQL and use function type parameters and anonymous functions. This way, you can define business logic more flexibly. You can use SQL functions to simplify feature implementation and improve code reuse. SQL functions provide the following features:

- SQL functions allow you to reference and call SQL UDFs in SQL scripts.
- You can specify built-in functions, UDFs, or SQL functions in function type parameters when you call SQL functions.
- You can specify anonymous functions in function type parameters when you call SQL functions.

Create a permanent SQL function

After you create a permanent SQL function and store it in the metadata system, all query statements can reference this function. To reference SQL functions, you must have function-level permissions. The following example shows the syntax:

```
CREATE SQL FUNCTION function_name(@parameter_in1 datatype[, @parameter_in2 datatype...])  
[RETURNS @parameter_out datatype]  
AS [BEGIN  
function_expression  
[END];
```

Parameters:

- `function_name`: the name of the SQL function that you want to create. Each function name must be unique and can be registered only once. The SQL function names cannot be the same as those of built-in functions.
- `parameter_in`: the input parameters of the SQL function.
- `RETURNS`: the variable to be returned by the SQL function. If you do not specify `RETURNS`, the value of `function_name` is returned.
- `parameter_out`: the output parameters of the SQL function.
- `function_expression`: the expression of the SQL function.

Example:

```
CREATE SQL FUNCTION MY_ADD(@a BIGINT) AS @a + 1;
```

In the preceding example, `@a + 1` indicates the implementation logic of the SQL function. You can write it as an expression to specify a built-in operator, built-in function, or UDF.

If the implementation logic is complex, you can write multiple SQL statements and enclose them with `BEGIN` and `END`. `RETURNS` specifies the variable to be returned by the SQL function. If you do not specify `RETURNS`, the value of `function_name` is returned.

Example:

```
CREATE SQL FUNCTION MY_SUM(@a BIGINT, @b BIGINT, @c BIGINT) RETURNS @my_sum BIGINT  
AS BEGIN  
    @temp := @a + @b;  
    @my_sum := @temp + @c;  
END;
```

Query an SQL function

You can query an SQL function in the same way as querying a Java or Python UDF. Example:

```
DESC FUNCTION my_add;
```

Delete an SQL function

You can delete an SQL function in the same way as deleting a Java or Python UDF. Example:

```
DROP FUNCTION my_add;
```

Call an SQL function

You can call an SQL function in the same way as calling an existing built-in function. Example:

```
SELECT my_sum(col1, col2, col3) from t;
```

Use temporary SQL functions

You can use temporary SQL functions if you no longer need to save SQL functions to the metadata system of MaxCompute. The temporary SQL functions are valid only in the current script. Example:

```
FUNCTION MY_ADD(@a BIGINT) AS @a + 1;
SELECT MY_ADD(key), MY_ADD(value) FROM src;
```

Use function type parameters

You can specify built-in functions, UDFs, or SQL functions in function type parameters when you call SQL functions. Example:

```
FUNCTION ADD(@a BIGINT) AS @a + 1;
FUNCTION OP(@a BIGINT, @fun FUNCTION (BIGINT) RETURNS BIGINT) AS @fun(@a);
SELECT OP(key, ADD), OP(key, ABS) FROM VALUES (1),(2) AS t (key);
```

Returned results:

```
+-----+-----+
|_c0  |_c1  |
+-----+-----+
| 2   | 1   |
| 3   | 2   |
+-----+-----+
```

In this example, two input parameters are specified for the OP function. The @a parameter specifies a value of the BIGINT type. The @fun parameter specifies a function whose input and output parameters are both of the BIGINT type. The OP function transfers @a to the function that is specified by @fun and then to the ADD and ABS functions for processing.

Use anonymous functions

You can specify anonymous functions in function type parameters when you call SQL functions. Example:

```
FUNCTION OP(@a BIGINT, @fun FUNCTION (BIGINT) RETURNS BIGINT) AS @fun(@a);
SELECT OP(key, FUNCTION (@a) AS @a + 1) FROM VALUES (1),(2) AS t (key);
```

In this example, FUNCTION (@a) AS @a + 1 is an anonymous function. You do not need to specify a data type for the input parameter @a. The compiler will infer the data type of @a based on the parameter definition of the OP function.

1.6.7.9. Embedded UDF

1.6.7.9.1. Background information

This topic describes the background information of code-embedded user-defined functions (UDFs).

The code-embedded UDFs of MaxCompute resolve the following issues in code implementation and maintenance:

- **Complicated code implementation:** After you create UDFs and develop code, you must compile the code in Java and create resources and functions.
- **Inconvenient code maintenance:** You cannot directly view the implementation logic of the UDFs that are referenced in SQL scripts or obtain the source code of JAR packages.
- **Poor code readability:** To implement Java library functions by using user-defined types (UDTs), you must write Java code as expressions in long code lines. In addition, you may fail to write some Java code as

expressions. Example:

```
Foo f = new Foo();  
f.execute();  
f.getResult();
```

1.6.7.9.2. Feature summary

This topic describes the features of code-embedded user-defined functions (UDFs).

Code-embedded UDFs allow you to embed Java or Python code into SQL scripts. When you compile a script, Janino-compiler identifies and extracts the embedded code, compiles the code in Java, and then dynamically generates resources and creates temporary functions.

You can place SQL scripts and third-party code lines in the same source code file. This simplifies the usage of user-defined types (UDTs) or UDFs and facilitates routine development and maintenance.

1.6.7.9.3. Limits

This topic describes the limits on code-embedded user-defined functions (UDFs).

You can use only Janino-compiler to compile embedded Java code. The syntax of the embedded Java code must be a subset of the standard JDK syntax. Embedded Java code has the following limits:

- Lambda expressions are not supported.
- You cannot specify multiple types of exceptions in a single catch block, for example, `catch(Exception1 | Exception2 e)`.
- Generics cannot be automatically inferred, for example, `Map map = new HashMap<>()`.
- Expressions for type parameter inference are ignored and you must use cast expressions to specify the parameter type, for example, `(String) myMap.get(key)`.
- Assertions are forcibly enabled, even if the `-ea` option of the Java Virtual Machine (JVM) is used.
- Code that is programmed in versions later than Java 8 is not supported.

1.6.7.9.4. Examples

This topic describes how to use code-embedded UDFs and provides examples for reference.

Reference embedded code in a UDT

Example:

```

SELECT
  s,
  com.mypackage.Foo.extractNumber(s)
FROM VALUES ('abc123def'),('apple') AS t(s);
#CODE ('lang='JAVA')
package com.mypackage;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
public class Foo {
  final static Pattern compile = Pattern.compile(".*?[0-9]+.*");
  public static String extractNumber(String input) {
    final Matcher m = compile.matcher(input);
    if (m.find()) {
      return m.group(1);
    }
    return null;
  }
}
#END CODE;

```

Description:

- #CODE indicates the beginning of the embedded code block. #END CODE indicates the end of the embedded code block. In this example, the embedded code block is placed at the end of the script and applies to the whole script.
- 'lang='JAVA' indicates that the embedded code is in Java. JAVA can be replaced with PYTHON if you compile code in Python.
- You can use UDT syntax in the SQL script to call Foo.extractNumber.

Define and call a Java code-embedded UDF

Example:

```

CREATE TEMPORARY FUNCTION foo AS 'com.mypackage.Reverse' USING
#CODE ('lang='JAVA')
package com.mypackage;
import com.aliyun.odps.udf.UDF;
public class Reverse extends UDF {
  public String evaluate(String input) {
    if (input == null) return null;
    StringBuilder ret = new StringBuilder();
    for (int i = input.toCharArray().length - 1; i >= 0; i--) {
      ret.append(input.toCharArray()[i]);
    }
    return ret.toString();
  }
}
#END CODE;
SELECT foo('abcd');

```

Description:

- You can place the embedded code block next to USING or at the end of the script. If you place it next to USING, it applies only to the CREATE TEMPORARY FUNCTION statement.
- The function created by CREATE TEMPORARY FUNCTION is a temporary function. This temporary function is executed only during the current execution process and is not stored in the MaxCompute meta system.

Define and call a Java code-embedded UDTF

Example:

```
CREATE TEMPORARY FUNCTION foo AS 'com.mypackage.Reverse' USING
#CODE ('lang'='JAVA', 'filename'='embedded.jar')
package com.mypackage;
import com.aliyun.odps.udf.UDTF;
import com.aliyun.odps.udf.UDFException;
import com.aliyun.odps.udf.annotation.Resolve;
@Resolve({"string->string,string"})
public class Reverse extends UDTF {
    @Override
    public void process(Object[] objects) throws UDFException {
        String str = (String) objects[0];
        String[] split = str.split(",");
        forward(split[0], split[1]);
    }
}
#END CODE;
SELECT foo('ab,dc') AS (a,b);
```

The return value of `@Resolve` must be of the `string[]` type. However, Janino-compiler cannot identify `"string->string,string"` in embedded code as `string[]`. To enable Janino-compiler to identify `"string->string,string"` as `string[]`, `"string->string,string"` must be enclosed in braces `{}`. If you create a Java UDTF by using a common method, braces `{}` are not required.

Define and call a Python code-embedded UDF

Example:

```
CREATE TEMPORARY FUNCTION foo AS 'embedded.UDFTest' USING
#CODE ('lang'='PYTHON', 'filename'='embedded')
from odps.udf import annotate
@annotate("bigint->bigint")
class UDFTest(object):
    def evaluate(self, a):
        return a * a
#END CODE;
SELECT foo(4);
```

Description:

- The indentation of Python code must comply with the specifications of the Python language.
- When you create a Python UDF, the class name that follows the AS clause must contain the file name of the Python source code. You can use `'filename'='embedded'` to specify a virtual file name.

1.6.8. UDTs

1.6.8.1. Scenarios and limits

This topic describes the scenarios and limits of user-defined types (UDTs).

MaxCompute introduces UDTs based on the new-generation SQL engine. UDTs allow you to reference classes or objects of third-party programming languages in SQL statements to call methods or obtain data.

UDTs are suitable for the following scenarios:

- You want to use some features that are not provided by MaxCompute but can be implemented in other programming languages.

For example, to implement some features, you need only to call built-in Java classes once. However, MaxCompute does not provide methods to implement these features. If you use user-defined functions (UDFs) to run these tasks, the procedure is complex.

- You want to call a third-party library in SQL statements to implement the related features. In this scenario, UDFs allow you to directly use a function provided by a third-party library in a SQL statement, instead of wrapping the function inside a UDF.
- You want to directly call the source code of a third-party programming language in SQL statements. The SELECT TRANSFORM statement allows you to write scripts to SQL statements. This improves readability and facilitates code maintenance. For some programming languages, such as Java, the source code can be executed only after it is compiled. You can use UDTs to reference objects and classes of these languages in SQL statements.

Limits

- UDTs support only Java. By default, all classes of SDK for Java can be referenced by UDTs.

 **Note** JDK 1.8 is used. A version later than JDK 1.8 may not be supported.

- All operators use the semantics of MaxCompute SQL instead of UDTs.
- UDTs cannot be used as shuffle keys in clauses, such as JOIN, GROUP BY, DISTRIBUTE BY, SORT BY, ORDER BY, or CLUSTER BY.
- DDL statements do not support UDTs. You cannot create tables that contain UDT objects. The final output cannot be UDT types.

1.6.8.2. Feature summary

This topic describes the features of user-defined types (UDTs) and provides examples for reference.

UDTs supported by many SQL engines are similar to the STRUCT type in MaxCompute. UDTs supported by MaxCompute are similar to the CREATE TYPE statement. A UDT contains both fields and methods. You do not need to use data definition language (DDL) statements to define new data types in MaxCompute. Instead, MaxCompute allows you to reference new data types directly in SQL statements. The following examples show how to use UDTs.

For example, to call the java.lang package in SQL statements, you can use one of the following methods:

- Use UDTs to call the java.lang package

```
-- Enable new data types. A new type of INTEGER (INT) is used in this example.
set odps.sql.type.system.odps2=true;
SELECT java.lang.Integer.MAX_VALUE;
```

Based on Java conventions, you can also omit the java.lang package from the preceding statement and use the following statement:

```
set odps.sql.type.system.odps2=true;
SELECT Integer.MAX_VALUE;
```

Returned results:

```
+-----+  
| max_value |  
+-----+  
| 2147483647 |  
+-----+
```

- Use user-defined functions (UDFs) to call the java.lang package
 - i. Define a UDF class.

```
package com.aliyun.odps.test;  
public class IntegerMaxValue extends com.aliyun.odps.udf.UDF {  
    public Integer evaluate() {  
        return Integer.MAX_VALUE;  
    }  
}
```

- ii. Compile the UDF into a JAR package, upload the package, and create a function.

```
add jar odps-test.jar;  
create function integer_max_value as 'com.aliyun.odps.test.IntegerMaxValue' using 'odps-test.jar';
```

- iii. Call the function in the SQL statement.

```
select integer_max_value();
```

In this example, UDTs simplify the procedure for you to use other programming languages to extend SQL features.

1.6.8.3. Implementation principles and feature description

This topic describes the implementation principles of user-defined types (UDTs) and their features.

Implementation principles

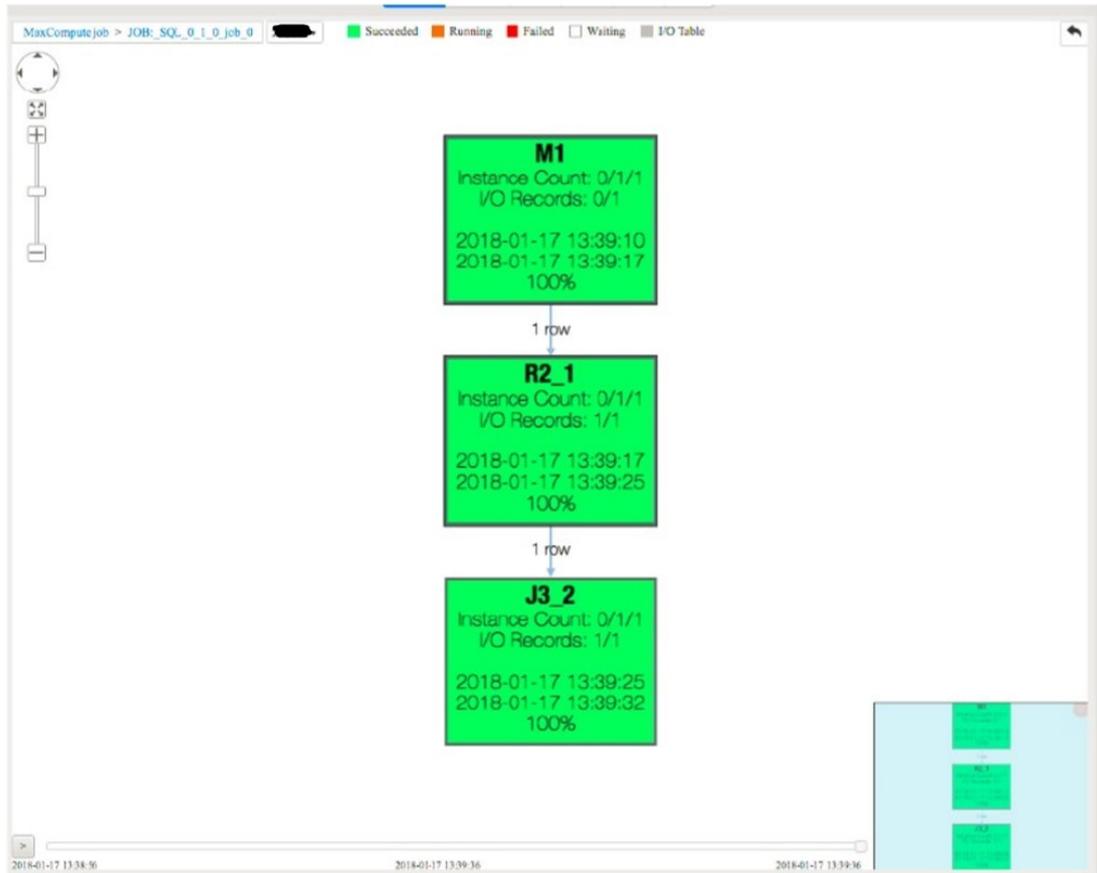
The following example shows how to run a UDT:

```
-- Sample data.  
@table1 := select * from values ('10000000000000000000') as t(x);  
@table2 := select * from values (100L) as t(y);  
-- Code logic.  
-- Create an object by using the new method.  
@a := select new java.math.BigInteger(x) x from @table1;  
-- Call a static method.  
@b := select java.math.BigInteger.valueOf(y) y from @table2;  
-- Call an instance method.  
select /*+mapjoin(b)*/ x.add(y).toString() from @a a join @b b;
```

Returned results:

```
1000000000000000000100
```

The following figure shows the process.



This UDT has three stages: M1, R2, and J3. If a JOIN operation is used in MapReduce, data must be reshuffled. As a result, data is processed at multiple stages. The processes and physical machines that process data vary at different stages.

Only the `new java.math.BigInteger(x)` method is called at the M1 stage.

The `java.math.BigInteger.valueOf(y)` and `x.add(y).toString()` methods are separately called at the J3 stage. These methods are called at different stages and executed in different processes and on different physical machines. UDTs encapsulate these stages to achieve an effect similar to all the stages being implemented on the same Java Virtual Machine (JVM).

The preceding example shows that the result of a subquery supports UDT columns. The `x` column retrieved by variable `a` is of the `java.math.BigInteger` type rather than a built-in type. You can transfer the UDT data to another operator and then call its method. You can also use the UDT data in a data shuffle.

Feature description

- UDTs support only Java. By default, all classes of SDK for Java can be referenced by UDTs.
- UDTs also allow you to upload JAR packages and directly reference these packages. Some flags are provided for UDTs.

- `set odps.sql.session.resources`: specifies the resource that you want to reference. Separate multiple resources with commas (,). For example, you can set this flag to `foo.sh,bar.txt`.

Note This flag works the same as the flag used to specify resources in the `SELECT TRANSFORM` statement. Therefore, this flag controls two features.

```
set odps.sql.type.system.odps2=true;
set odps.sql.session.resources=odps-test.jar;
-- Specify the JAR package that you want to reference. You must upload the package to your project in advance.
select new com.aliyun.odps.test.IntegerMaxValue().evaluate();
```

- `odps.sql.session.java.imports`: specifies the default Java package. You can specify multiple packages and separate them with commas (,). This flag is similar to the `IMPORT` statement in Java. You can specify a class path, such as `java.math.BigInteger`, or use asterisks (*). `static import` is not supported.

```
set odps.sql.type.system.odps2=true;
set odps.sql.session.resources=odps-test.jar;
set odps.sql.session.java.imports=com.aliyun.odps.test.*;
-- Specify the default JAR package.
select new IntegerMaxValue().evaluate();
```

- UDTs support resource access. In MaxCompute SQL, you can call the static method `com.aliyun.odps.udf.impl.UDTExecutionContext.get()` to obtain the `ExecutionContext` object. Then, you can use this object to access the current `ExecutionContext` class and then access resources, such as files or tables.
- UDTs support the following operations:
 - Create objects by using the `new` method.
 - Create arrays by using the `new` method. Initializer lists can be used. Example: `new Integer[] { 1, 2, 3 }`.
 - Call methods, including static methods.

Note

- The identifiers in UDTs contain the names of packages, classes, methods, and fields. All identifiers are case-sensitive.
- Anonymous classes and lambda expressions are not supported.
- UDTs are used in expressions. Functions that do not return values cannot be called in expressions. This issue will be resolved in later versions.

- UDTs support the following data types:
 - UDTs support SQL type conversions, such as `cast(1 as java.lang.Object)`. UDTs do not support Java type conversions, such as `(Object)1`.

- The built-in types of UDTs have a one-to-one mapping relationship with specific Java types.
 - You can directly call the method of the Java type to which the built-in type is mapped. Example: `'123'.length(), 1L.hashCode()`.
 - UDTs can be used in built-in functions and UDFs. For example, in `chr(Long.valueOf('100'))`, `Long.valueOf` returns a value of the `java.lang.Long` type. The built-in function `CHR` supports the built-in `BIGINT` type.
 - The data of a Java primitive type is automatically converted to the boxing type and the preceding two rules apply.

Note For some new built-in data types, you must use `set odps.sql.type.system.odps2=true;` to declare these types. Otherwise, an error occurs.

- The following type conversion rules apply in UDTs:
 - UDT objects can be implicitly converted to the objects of their base classes.
 - UDT objects can be forcibly converted to the objects of their base classes or subclasses.
 - The data type conversion between two objects without inheritance follows the original conversion rules. However, such conversions may result in changes to the data. For example, data of the `java.lang.Long` type can be forcibly converted to the `java.lang.Integer` type. This conversion uses the rules that are used to convert the built-in `BIGINT` type to the `INT` type. This process may result in changes to the data and even loss of data precision.
- UDTs support Java generics. For example, the compiler can determine that the value returned by `java.util.Arrays.asList(new java.math.BigInteger('1'))` is of the `java.util.List<java.math.BigInteger>` type based on the parameter type.

Note You must specify the type parameter in a constructor function or use `java.lang.Object`. This is the same as Java.

- All operators use the semantics of MaxCompute SQL. Examples:
 - Combination of strings: The result of `String.valueOf(1) + String.valueOf(2)` is 3. The two strings are implicitly converted to `DOUBLE`-type values and summed. If you use Java string concatenation to combine the strings, the result is 12.
 - = operations: The = operator in SQL statements is used as a comparison operator. It is used to compare one expression with another. You must call the `equals` method in Java to check whether two objects are equivalent. The = operator cannot be used to verify the equivalence of two objects.
- UDTs do not have a clear definition of object equality. This is caused by data reshuffling. Objects may be transmitted between different processes or physical machines. During object transmission, an object may be referenced as two different objects. For example, an object may be shuffled to two machines and then reshuffled. Therefore, when you use UDTs, you must use the `equals` method instead of the = operator to verify the equivalence of two objects.

Objects in the same row or column are correlated in some way. However, a correlation between objects in different rows or columns cannot be ensured.
- UDTs cannot be used as shuffle keys in clauses, such as `JOIN`, `GROUP BY`, `DISTRIBUTE BY`, `SORT BY`, `ORDER BY`, or `CLUSTER BY`.
- UDTs can be used at the stages of expressions, but cannot be used as outputs.
- You can use UDTs to implement the feature provided by the `SCALAR` function. You can use the built-in functions `COLLECT_SET` and `EXPLODE` with UDTs to implement the features provided by aggregate and table-valued functions.

1.6.8.4. Feature advantages

This topic describes the feature advantages of user-defined types (UDTs).

UDTs have the following feature advantages:

- UDTs are easy to use. You do not need to define functions.
- UDTs support all JDK features. This improves SQL flexibility.
- UDT code can be stored in the same file as the SQL code. This facilitates code management.
- You can directly reference the libraries of other programming languages and reuse code that you have written in other languages.
-

1.6.8.5. Performance advantages

This topic describes the performance advantages of user-defined types (UDTs).

UDTs run in a similar way to UDFs in terms of performance. Therefore, the performance of UDTs is almost the same as that of UDFs. The optimized computing engine improves the performance of UDTs in specific scenarios.

- If a UDT object is used in different processes, it must be serialized and deserialized. If you use a UDT to perform operations that do not require data reshuffling, such as JOIN or AGGREGATE, the overheads of serialization and deserialization are avoided.
- The runtime of UDTs is based on Codegen rather than reflection. Therefore, no performance loss occurs. Multiple UDTs can be executed in a single function call. In the following example, it seems that a UDT is called multiple times, but the UDF is actually called only once. Therefore, no additional interface overheads are caused even though the operational units of UDTs are small.

```
values[x].add(values[y]).divide(java.math.BigInteger.valueOf(2))
```

1.6.8.6. Security advantages

This topic describes the security advantages of user-defined types (UDTs).

Similar to UDFs, UDTs are restricted in the Java sandbox model. To perform restricted operations, you must cancel sandbox isolation for the operations or apply to join a sandbox whitelist.

1.6.8.7. More examples

1.6.8.7.1. Example of using Java arrays

This topic provides an example of using Java arrays of user-defined types (UDTs).

Example:

```
set odps.sql.type.system.odps2=true;
set odps.sql.udt.display.toString=true;
SELECT
  new Integer[10], -- Create an array that contains 10 elements.
  new Integer[] {c1, c2, c3}, -- Initialize an ArrayList to create an array that contains three elements.
  new Integer[][] { new Integer[] {c1, c2}, new Integer[] {c3, c4} }, -- Create a multidimensional array.
  new Integer[] {c1, c2, c3} [2], -- Access the elements in the array by using indexes.
  java.util.Arrays.asList(c1, c2, c3); -- Create a list of the List<Integer> type, which can be used as an array of the Array<Int> type. This is another way to create a built-in array.
FROM VALUES (1,2,3,4) AS t(c1, c2, c3, c4);
```

1.6.8.7.2. Example of using JSON

This topic provides an example of using JSON for user-defined types (UDTs).

The runtime of a UDT carries a JSON dependency (V2.2.4), which can be directly used in JSON.

Note In addition to JSON dependencies, MaxCompute runtime also carries other dependencies, including commons-logging (1.1.1), commons-lang (2.5), commons-io (2.4), and protobuf-java (2.4.1).

Example:

```
set odps.sql.type.system.odps2=true;
set odps.sql.session.java.imports=java.util.*,java.com.google.gson.*; -- To import multiple packages at a time, separate the packages with commas (,).
@a := select new Gson() gson; -- Create a Gson object.
select
  gson.toJson(new ArrayList<Integer>(Arrays.asList(1, 2, 3))), -- Convert an object to a JSON string.
  cast(gson.fromJson("[\"a\",\"b\",\"c\"]", List.class) as List<String>) -- Deserialize the JSON string. Gson forcibly converts the deserialization result from the List<Object> type to the List<String> type.
from @a;
```

Returned results:

```
+-----+-----+
|_c0   |_c1   |
+-----+-----+
|[1,2,3]| 1     |
+-----+-----+
```

Compared with the built-in function GET_JSON_OBJECT, this UDT-based method is simpler. In addition, this method deserializes the content that is extracted from the JSON string to a supported data type. The deserialization improves efficiency.

1.6.8.7.3. Example of using complex data types

This topic provides an example of using complex data types of user-defined types (UDTs).

The built-in data type ARRAY maps the java.util.List method, and the built-in data type MAP maps the java.util.Map method.

- Java objects in classes that implement java.util.List or java.util.Map can be used to process complex-type data in MaxCompute SQL.
- MaxCompute can directly call java.util.List or java.util.Map to process data of the ARRAY or MAP type.

Example:

```

set odps.sql.type.system.odps2=true;
set odps.sql.session.java.imports=java.util.*;
select
  size(new ArrayList<Integer>()), -- Call the built-in function size() to obtain the size of the ArrayList.
  array(1,2,3).size(),          -- Call the built-in function size() of the java.util.List method for the built-in data type ARRAY.
  sort_array(new ArrayList<Integer>()), -- Sort the data in the ArrayList.
  al[1],                          -- java.util.List does not support indexing but can process data of the ARRAY type that supports indexing.
  Objects.toString(a),            -- Convert data from the ARRAY type to the STRING type.
  array(1,2,3).subList(1, 2)      -- Obtain a sublist.
from (select new ArrayList<Integer>(array(1,2,3)) as al, array(1,2,3) as a) t;

```

Returned results:

```

+-----+-----+-----+-----+-----+-----+
|_c0  |_c1  |_c2  |_c3  |_c4  |_c5  |
+-----+-----+-----+-----+-----+
| 0   | 3   | []  | 2   | [1, 2, 3] | [2]  |
+-----+-----+-----+-----+-----+

```

1.6.8.7.4. Aggregation example

This topic provides an example of using user-defined types (UDTs) to aggregate data.

To aggregate data by using a UDT, you must use the built-in function COLLECT_SET or COLLECT_LIST to aggregate data to a list and then call the UDT to calculate the aggregate value.

The following example shows how to calculate the median of BigInteger data. You cannot directly call the built-in function MEDIAN because the data is of the java.math.BigInteger type.

```

set odps.sql.session.java.imports=java.math.*;
@test_data := select * from values (1),(2),(3),(5) as t(value);
@a := select collect_list(new BigInteger(value)) values from @test_data; -- Aggregate the data to a list.
@b := select sort_array(values) as values, values.size() cnt from @a; -- Sort the data.
@c := select if(cnt % 2 == 1, new BigDecimal(values[cnt div 2]), new BigDecimal(values[cnt div 2 - 1].add(values[cnt div 2])).divide(new BigDecimal(2))) med from @b;
-- Obtain the final output.
select med.toString() from @c;

```

Returned results:

```

+-----+
|_c0  |
+-----+
| 2.5 |
+-----+

```

The COLLECT_LIST function cannot be used to aggregate partial data because it can only aggregate all data in a specific group. It is less efficient than the built-in aggregate functions of MaxCompute or user-defined aggregate functions (UDAFs). We recommend that you use built-in aggregate functions if possible. If all data in a group is aggregated, data skew may occur.

The UDT-based method produces a higher efficiency than UDAFs or built-in aggregate functions, such as WM_CONCAT, for aggregating all data in a group.

1.6.8.7.5. Example of using table-valued functions

This topic provides an example of table-valued functions of user-defined types (UDTs).

Table-valued functions allow you to specify multiple input rows and columns, and can generate multiple output rows and columns. To implement a table-valued function, perform the following steps:

- Specify multiple input rows or columns. For more information, see [Example of aggregation](#).
- Call the `java.util.List` or `java.util.Map` method to generate a data collection, and then call the `explode` function to split the collection into multiple rows.
- Call different getter methods to obtain data from different fields in a UDT. The obtained data is returned in multiple columns.

The following example shows how to expand a JSON string.

```
@a := select '[{"a": "1", "b": "2"}, {"a": "1", "b": "2"}]' str; -- The sample data.
@b := select new com.google.gson.Gson().fromJson(str, java.util.List.class) l from @a; -- Deserialize the JSON string
.
@c := select cast(e as java.util.Map<Object, Object>) m from @b lateral view explode(l) t as e; -- Call the explode function to split the string.
@d := select m.get('a') as a, m.get('b') as b from @c; -- Return the splitting result in multiple columns.
select a.toString() a, b.toString() b from @d; -- The final output. Columns a and b in the variable d are of the Object type.
```

Returned results:

```
+-----+-----+
| a   | b   |
+-----+-----+
| 1   | 2   |
+-----+-----+
| 1   | 2   |
+-----+-----+
```

1.6.9. UDJ

1.6.9.1. Background information

This topic describes the background information of user-defined join (UDJ).

MaxCompute provides multiple JOIN methods, including INNER JOIN, RIGHT JOIN, OUTER JOIN, LEFT JOIN, FULL JOIN, SEMI JOIN, and ANTI-SEMI JOIN. You can use these built-in JOIN methods in most scenarios. However, these methods are insufficient if you want to perform cross join operations.

In most cases, you can use user-defined functions (UDFs) to describe your code framework. However, the current UDF, user-defined table-valued function (UDTF), and user-defined aggregate function (UDAF) frameworks can only handle one table at a time. To perform user-defined operations for multiple tables, you must use built-in JOIN methods, UDFs, UDTFs, and complex SQL statements. In such scenarios, you must use a custom MapReduce framework instead of SQL to complete the required computing tasks.

Regardless of the scenario, these operations require technological expertise and may cause the following issues:

- In scenarios where you use built-in JOIN methods, UDFs, UDTFs, and complex SQL statements: The use of multiple JOIN methods and code in SQL statements results in a logical black box, which makes it difficult to generate an optimal execution plan.

- In scenarios where you use a custom MapReduce framework: Execution plans are hard to optimize. Most of the MapReduce code is written in Java. During the deep optimization of native runtime code, the execution of the MapReduce code is less efficient than the execution of the MaxCompute code that is generated by the LLVM code generator.

MaxCompute introduces UDJ to the UDF framework based on the MaxCompute V2.0 computing engine. You can flexibly join tables by using UDJ to perform more customized operations. This simplifies MapReduce-based underlying operations in the distributed system.

1.6.9.2. Examples of using UDJ

1.6.9.2.1. Cross join operation by using UDJ

This topic describes how to use user-defined join (UDJ) to perform cross join operations and provides an example for reference.

Assume that two log tables named `payment` and `user_client_log` exist.

- The `payment` table stores the payment records of users. Each payment record contains the user ID, payment time, and payment content. The following table lists the sample data.

user_id	time	pay_info
2656199	2018-02-13 22:30:00	gZhvdYSOQb
8881237	2018-02-13 08:30:00	pYvotuLDIT
8881237	2018-02-13 10:32:00	KBuMzRpsko

- The `user_client_log` table stores the client logs of users. Each log contains the user ID, logging time, and log content. The following table lists the sample data.

user_id	time	content
8881237	2018-02-13 00:30:00	click MpkvilgWSmhUuPn
8881237	2018-02-13 06:14:00	click OkTYNUHMqZzIDyL
8881237	2018-02-13 10:30:00	click OkTYNUHMqZzIDyL

Requirement: For each record in the `user_client_log` table, find the payment record that has the closest time to this record in the `payment` table. Then, join the two records and generate results. The following table lists the results.

user_id	time	content
8881237	2018-02-13 00:30:00	click MpkvilgWSmhUuPn, pay pYvotuLDIT
8881237	2018-02-13 06:14:00	click OkTYNUHMqZzIDyL, pay pYvotuLDIT
8881237	2018-02-13 10:30:00	click OkTYNUHMqZzIDyL, pay KBuMzRpsko

To meet this requirement, use one of the following methods:

- Use built-in JOIN methods. SQL sample code:

```
SELECT
  p.user_id,
  p.time,
  merge(p.pay_info, u.content)
FROM
  payment p RIGHT OUTER JOIN user_client_log u
ON p.user_id = u.user_id and abs(p.time - u.time) = min(abs(p.time - u.time))
```

If you join two rows in the tables, you must calculate the minimum difference between the p.time and u.time under the same user_id. However, you cannot call aggregate functions in the join condition. Therefore, you cannot use standard JOIN methods to complete this task.

- Use the UDJ method.
 - Create a UDJ function.
 - Configure the SDK of the new version.

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>odps-sdk-udf</artifactId>
  <version>0.30.0</version>
  <scope>provided</scope>
</dependency>
```

- Write UDJ code and package the code as odps-udj-example.jar.

```
package com.aliyun.odps.udf.example.udj;
import com.aliyun.odps.Column;
import com.aliyun.odps.OdpsType;
import com.aliyun.odps.Yieldable;
import com.aliyun.odps.data.ArrayRecord;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.udf.DataAttributes;
import com.aliyun.odps.udf.ExecutionContext;
import com.aliyun.odps.udf.UDJ;
import com.aliyun.odps.udf.annotation.Resolve;
import java.util.ArrayList;
import java.util.Iterator;
/** For each record in the table on the right, find the record in the table on the left that has the closest time to this record.
 * Join the two records.
 */
@Resolve("->string,bigint,string")
public class PayUserLogMergeJoin extends UDJ {
  private Record outputRecord;
  /** Call the preceding code before data is processed. You can use the PayUserLogMergeJoin method for initialization.
   */
  @Override
  public void setup(ExecutionContext executionContext, DataAttributes dataAttributes) {
    //
    outputRecord = new ArrayRecord(new Column[]{
      new Column("user_id", OdpsType.STRING),
      new Column("time", OdpsType.BIGINT),
      new Column("content", OdpsType.STRING)
    });
  }
}
```

```
/** Rewrite the PayUserLogMergeJoin method to implement the connection logic.
 * @param key: the current key for joining.
 * @param left: the record group of the current key in the table on the left.
 * @param right: the record group of the current key in the table on the right.
 * @param output: generates the result of UDJ.
 */
@Override
public void join(Record key, Iterator<Record> left, Iterator<Record> right, Yieldable<Record> output) {
    outputRecord.setString(0, key.getString(0));
    if (!right.hasNext()) {
        // The group on the right is empty. Do not perform operations on this group.
        return;
    } else if (!left.hasNext()) {
        // The group on the left is empty. Generate all records from the group on the right but do not join the
        records of the two tables.
        while (right.hasNext()) {
            Record logRecord = right.next();
            outputRecord.setBigint(1, logRecord.getDatetime(0).getTime());
            outputRecord.setString(2, logRecord.getString(1));
            output.yield(outputRecord);
        }
        return;
    }
    ArrayList<Record> pays = new ArrayList<>();
    // The records in the group on the left are iterated from the beginning to the end.
    // The iterator cannot reset the records in the group on the right.
    // Save each record of the group on the left to ArrayList.
    left.forEachRemaining(pay -> pays.add(pay.clone()));
    while (right.hasNext()) {
        Record log = right.next();
        long logTime = log.getDatetime(0).getTime();
        long minDelta = Long.MAX_VALUE;
        Record nearestPay = null;
        // Iterate all the records of the group on the left. Then, you can find the minimum time difference bet
        ween the records.
        for (Record pay: pays) {
            long delta = Math.abs(logTime - pay.getDatetime(0).getTime());
            if (delta < minDelta) {
                minDelta = delta;
                nearestPay = pay;
            }
        }
        // Merge the log records and payment records that are closest in time and then generate results.
        outputRecord.setBigint(1, log.getDatetime(0).getTime());
        outputRecord.setString(2, mergeLog(nearestPay.getString(1), log.getString(1)));
        output.yield(outputRecord);
    }
}
String mergeLog(String payInfo, String logContent) {
    return logContent + ", pay " + payInfo;
}
@Override
public void close() {
}
}
```

- c. Add the odps-udj-example.jar package to MaxCompute.

```
add jar odps-udj-example.jar;
```

- d. Register the UDF function pay_user_log_merge_join in MaxCompute.

```
create function pay_user_log_merge_join
as 'com.aliyun.odps.udf.example.udj.PayUserLogMergeJoin'
using 'odps-udj-example.jar';
```

- ii. Prepare sample data.

- a. Create the payment table and the user_client_log table.

```
create table payment(user_id string,time datetime,pay_info string);
create table user_client_log(user_id string,time datetime,content string);
```

- b. Insert data into the sample tables.

```
-- Insert data into the payment table.
INSERT OVERWRITE TABLE payment VALUES
('1335656', datetime '2018-02-13 19:54:00', 'PEqMSHyktn'),
('2656199', datetime '2018-02-13 12:21:00', 'pYvotuLDIT'),
('2656199', datetime '2018-02-13 20:50:00', 'PEqMSHyktn'),
('2656199', datetime '2018-02-13 22:30:00', 'gZhvdYSOQB'),
('8881237', datetime '2018-02-13 08:30:00', 'pYvotuLDIT'),
('8881237', datetime '2018-02-13 10:32:00', 'KBuMzRpsko'),
('9890100', datetime '2018-02-13 16:01:00', 'gZhvdYSOQB'),
('9890100', datetime '2018-02-13 16:26:00', 'MxONdLckwa')
;

-- Insert data into the user_client_log table.
INSERT OVERWRITE TABLE user_client_log VALUES
('1000235', datetime '2018-02-13 00:25:36', 'click FNOXAibRjklAQPB'),
('1000235', datetime '2018-02-13 22:30:00', 'click GczrYaxvkiPultZ'),
('1335656', datetime '2018-02-13 18:30:00', 'click MxONdLckpAFUHRS'),
('1335656', datetime '2018-02-13 19:54:00', 'click mKRPGOCiFDyzTgM'),
('2656199', datetime '2018-02-13 08:30:00', 'click CZwafHsbJOPNitL'),
('2656199', datetime '2018-02-13 09:14:00', 'click nYHJqIjpvkKToy'),
('2656199', datetime '2018-02-13 21:05:00', 'click gbAfPCwrGXvEjpl'),
('2656199', datetime '2018-02-13 21:08:00', 'click dhpZyWwMuGjBOTJP'),
('2656199', datetime '2018-02-13 22:29:00', 'click bAsxnUdDhvfqaBr'),
('2656199', datetime '2018-02-13 22:30:00', 'click XlhZdLaOocQRmrY'),
('4356142', datetime '2018-02-13 18:30:00', 'click DYqShmGbloWKier'),
('4356142', datetime '2018-02-13 19:54:00', 'click DYqShmGbloWKier'),
('8881237', datetime '2018-02-13 00:30:00', 'click MpkvilgWSmhUuPn'),
('8881237', datetime '2018-02-13 06:14:00', 'click OkTYNUHMqZzldyL'),
('8881237', datetime '2018-02-13 10:30:00', 'click OkTYNUHMqZzldyL'),
('9890100', datetime '2018-02-13 16:01:00', 'click vOTQfBFjcgXisYU'),
('9890100', datetime '2018-02-13 16:20:00', 'click WxaLgOCCvEvhifJ')
;
```

- iii. Use a UDF function in SQL.

```
SELECT r.user_id, from_unixtime(time/1000) as time, content FROM (  
  SELECT user_id, time as time, pay_info FROM payment  
  ) p JOIN (  
  SELECT user_id, time as time, content FROM user_client_log  
  ) u  
  ON p.user_id = u.user_id  
  USING pay_user_log_merge_join(p.time, p.pay_info, u.time, u.content)  
  r  
  AS (user_id, time, content);
```

Parameters in the USING clause:

- `pay_user_log_merge_join`: the name of the UDJ function in SQL.
- `(p.time, p.pay_info, u.time, u.content)`: the columns in both of the tables used in the UDJ function.
- `r`: the alias of the result returned by the UDJ function. You can reference this alias in other SQL statements.
- `(user_id, time, content)` are the columns returned by the UDJ function.

Returned results:

```
+-----+-----+-----+  
| user_id | time   | content |  
+-----+-----+-----+  
| 1000235 | 2018-02-13 00:25:36 | click FNOXAibRjklQPB |  
| 1000235 | 2018-02-13 22:30:00 | click GczrYaxvkiPultZ |  
| 1335656 | 2018-02-13 18:30:00 | click MxONdLckpAFUHRS, pay PEqMSHyktn |  
| 1335656 | 2018-02-13 19:54:00 | click mKRPGOciFDyzTgM, pay PEqMSHyktn |  
| 2656199 | 2018-02-13 08:30:00 | click CZwafHsbJOPNitL, pay pYvotuLDIT |  
| 2656199 | 2018-02-13 09:14:00 | click nYHJqlpjevkKToy, pay pYvotuLDIT |  
| 2656199 | 2018-02-13 21:05:00 | click gbAfPCwrGXvEjpl, pay PEqMSHyktn |  
| 2656199 | 2018-02-13 21:08:00 | click dhpZyWMuGjBOTJP, pay PEqMSHyktn |  
| 2656199 | 2018-02-13 22:29:00 | click bAsxnUdDhvfqaBr, pay gZhvdySOQb |  
| 2656199 | 2018-02-13 22:30:00 | click XlhZdLaOocQRmrY, pay gZhvdySOQb |  
| 4356142 | 2018-02-13 18:30:00 | click DYqShmGbloWKier |  
| 4356142 | 2018-02-13 19:54:00 | click DYqShmGbloWKier |  
| 8881237 | 2018-02-13 00:30:00 | click MpkvilgWSmhUuPn, pay pYvotuLDIT |  
| 8881237 | 2018-02-13 06:14:00 | click OkTYNUHMqZzlDyL, pay pYvotuLDIT |  
| 8881237 | 2018-02-13 10:30:00 | click OkTYNUHMqZzlDyL, pay KBUmzRpsko |  
| 9890100 | 2018-02-13 16:01:00 | click vOTQfBFjcgXisYU, pay gZhvdySOQb |  
| 9890100 | 2018-02-13 16:20:00 | click WxaLgOCCvEvhfJ, pay MxONdLckwa |  
+-----+-----+-----+
```

1.6.9.2.2. Pre-sorting

This topic describes the pre-sorting feature of user-defined join (UDJ) and provides an example for reference.

An iterator is used to iterate all records in the payment table and find the payment record that has the closest time to a specific log record in the user_client_log table. To perform this task, you must load all payment records with the same user_id to an ArrayList. This method can be applied when the number of payment records is small. Due to the limits of the memory size, you must find another method to load the data if a large number of payment records have been generated. This topic describes how to address this issue by using the SORT BY clause.

If the number of payment records is too large to be stored in the memory and all the data in the table has been sorted by time, you need only to compare the first element in the two lists.

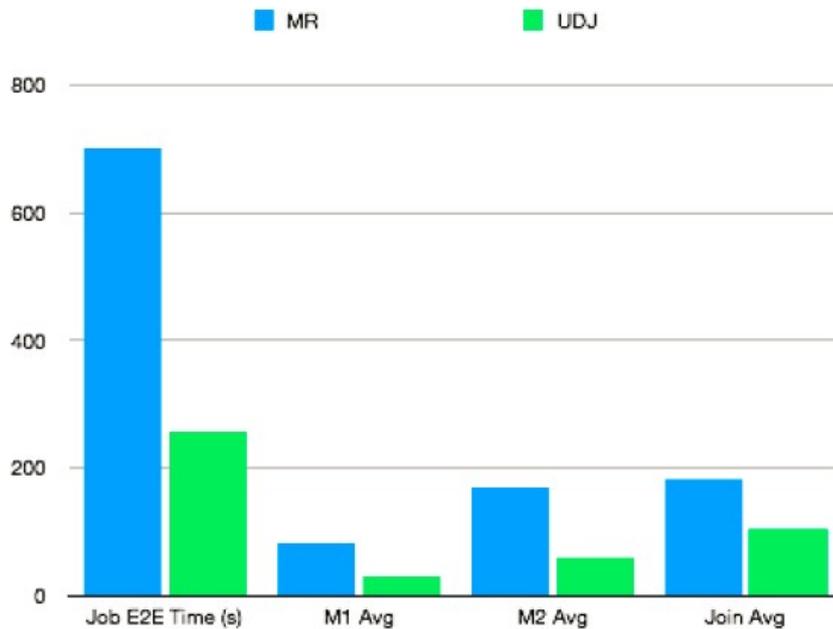
This method uses the SORT BY clause to pre-sort the UDJ data. To achieve the same effect by using the previous method, you need only to cache a maximum of three data records during pre-sorting. The following code shows an example:

```
@Override
public void join(Record key, Iterator<Record> left, Iterator<Record> right, Yieldable<Record> output) {
    outputRecord.setString(0, key.getString(0));
    if (! right.hasNext()) {
        return;
    } else if (! left.hasNext()) {
        while (right.hasNext()) {
            Record logRecord = right.next();
            outputRecord.setBigint(1, logRecord.getDatetime(0).getTime());
            outputRecord.setString(2, logRecord.getString(1));
            output.yield(outputRecord);
        }
        return;
    }
    long prevDelta = Long.MAX_VALUE;
    Record logRecord = right.next();
    Record payRecord = left.next();
    Record lastPayRecord = payRecord.clone();
    while (true) {
        long delta = logRecord.getDatetime(0).getTime() - payRecord.getDatetime(0).getTime();
        if (left.hasNext() && delta > 0) {
            // The time delta between the two records decreases and the operation can continue.
            // Explore the group on the left to try to obtain a smaller time delta.
            lastPayRecord = payRecord.clone();
            prevDelta = delta;
            payRecord = left.next();
        } else {
            // The minimum time delta point is reached. Check the final record in the payment table.
            // Generate the merged result and prepare to process the next record.
            // The group on the right.
            Record nearestPay = Math.abs(delta) < prevDelta ? payRecord : lastPayRecord;
            outputRecord.setBigint(1, logRecord.getDatetime(0).getTime());
            String mergedString = mergeLog(nearestPay.getString(1), logRecord.getString(1));
            outputRecord.setString(2, mergedString);
            output.yield(outputRecord);
            if (right.hasNext()) {
                logRecord = right.next();
                prevDelta = Math.abs(
                    logRecord.getDatetime(0).getTime() - lastPayRecord.getDatetime(0).getTime()
                );
            } else {
                break;
            }
        }
    }
}
```

1.6.9.3. Performance

This topic describes the performance of user-defined join (UDJ).

A real online MapReduce job is used as an example to verify the performance of UDJ. The job runs based on a complex algorithm. In this example, two tables are joined, UDJ is used to rewrite the MapReduce job, and the correctness of the UDJ results is checked. The following figure shows the MapReduce and UDJ performance under the same data concurrency.



As shown in the figure, UDJ conveniently describes the complex logic of how to handle multiple tables and greatly improves performance. The code is called only within UDJ. The entire Mapper logic in this example is executed by the native runtime engine of MaxCompute. The data exchange logic between the MaxCompute UDJ runtime engine and Java interfaces is optimized in Java code. The JOIN logic of UDJ is more efficient than that of a reducer.

1.6.10. Parameterized view

This topic describes the parameterized view feature that is supported by the MaxCompute SQL engine.

In the traditional views of MaxCompute, complex SQL scripts are encapsulated at the underlying layer. Callers can call views the same way as reading a common table without the need to understand the underlying implementation. Traditional views are widely used because they implement encapsulation and reuse. However, traditional views cannot accept parameters from callers. This reduces code reuse efficiency. For example, a caller cannot filter data in the underlying table that is read by a view nor pass other parameters. The MaxCompute SQL engine supports parameterized views and allows you to import tables or variables to customize views.

Define a parameterized view

You can use the following syntax to create a parameterized view:

```
-- view with parameters
-- param @a -a table parameter
-- param @b -a string parameter
-- returns a table with schema (key STRING,value STRING)
CREATE VIEW IF NOT EXISTS pv1(@a table (k STRING,v BIGINT), @b STRING)
AS
SELECT srcp.key,srcp.value FROM srcp JOIN @a ON srcp.key=a.k AND srcp.p=@b;
```

Syntax description:

- To create a parameterized view, you must specify parameters. Therefore, you must use Script Mode SQL to create a parameterized view.
- The created view pv1 has two parameters: TABLE and STRING. The parameter value can be a table or of a basic data type.
- The parameter value can also be a subquery, for example, `SELECT * FROM view_name((SELECT 1 FROM src WHERE a > 0), 1);`
- When you create a view, you can set ANY for a parameter, which indicates any data type. For example, `CREATE VIEW paramed_view (@a ANY) AS SELECT * FROM src WHERE CASE WHEN @a IS NULL THEN key1 ELSE key2 END = key3;` defines that the first parameter of the view can be a value of any data type.

However, the ANY type cannot be used in a specific operation, such as + or AND, which requires specific data types. A field of the ANY type is often used as a Passthrough column in the TABLE parameter.

Example:

```
CREATE VIEW paramed_view (@a TABLE(name STRING, id ANY, age BIGINT)) AS SELECT * FROM @a WHERE name = 'foo' AND age < 25;
-- The call example.
SELECT * FROM param_view((SELECT name, id, age FROM students));
```

Note

- After you execute the `CREATE VIEW` statement to create a view, you can run the `DESC` command to obtain the description of the view. This description contains the return type of the view.
- The return type of a view is recalculated when the view is called. It may be different from the return type, such as ANY, that you specify when you create the view.

- When you create a view, you can use an asterisk (*) in the TABLE parameter to retrieve any columns. The asterisk (*) can represent a specific data type or the ANY type. Example:

```
CREATE VIEW paramed_view (@a TABLE(key STRING, * ANY), @b TABLE(key STRING, * STRING)) AS SELECT a. * FROM @a JOIN @b ON a.key = b.key;
-- The call example.
SELECT name, address FROM param_view((SELECT school, name, age, address FROM student), school) WHERE age < 20;
```

In this example, the view accepts two TABLE parameters. In the table that is specified by the first TABLE parameter, data in the first column is of the STRING type, and data in other columns can be of the ANY type. In the table that is specified by the second TABLE parameter, data in the first and other columns is all of the STRING type. Take note of the following points:

- The varied-length part must be placed at the end of the table that is specified by the TABLE parameter. This means that the * column cannot be followed by other columns. Therefore, the table that is specified by the TABLE parameter can contain only one varied-length column.
- The varied-length part must be placed at the end of the table that is specified by the TABLE parameter. However, the columns of an input table may not be arranged in this sequence. In this case, the columns need to be rearranged. A subquery can be used as a parameter value and must be enclosed in a pair of parentheses ().
- No name is specified for the varied-length part of the table that is specified by the TABLE parameter. As a result, data in the varied-length part cannot be referenced or used for operation when you define the view.
- Although you cannot use the varied-length part for operation, you can use the `SELECT *` statement to transfer data in the varied-length part out of the table.

- The column of the table that is specified by the TABLE parameter may be different from the fixed-length column that you specify when you define the view. If the names are different, the compiler automatically renames the column of the table that is specified by the TABLE parameter. If the data types are different, the compiler performs an implicit conversion. If the implicit conversion fails, an error occurs.

Call a parameterized view

You can execute the following statement to call the pv1 view that you defined:

```
@a := SELECT * FROM src WHERE value >0;
--call view with table variable and scalar
@b := SELECT * FROM pv1(@a,'20170101');
@another_day := '20170102';
--call view with table name and scalar variable
@c := SELECT * FROM pv1(src2, @another_day);
@d := SELECT * FROM @c UNION ALL SELECT * FROM @b;
WITH
t AS(SELECT * FROM src3)
SELECT * FROM @c
UNION ALL
SELECT * FROM @d
UNION ALL
SELECT * FROM pv1(t,@another_day);
```

 **Note** You can use different parameters to call the pv1 view.

- The value of the TABLE parameter can be the name of a physical table, view, or table variable. You can also set the TABLE parameter to a table alias by using common table expressions (CTEs).
- The values of common parameters can be variables or constants.

Usage notes

- The script of a parameterized view can contain only data manipulation language (DML) statements. The statements, such as INSERT, CREATE TABLE, or PRINT, cannot be contained.
- A parameterized view can contain multiple SQL statements.

```
-- view with parameters
-- param @a -a table parameter
-- param @b -a string parameter
-- returns a table with schema (key string,value string)
CREATE VIEW IF NOT EXISTS pv2(@a TABLE (k STRING,v BIGINT), @b STRING) AS
BEGIN
@srcp := SELECT * FROM srcp WHERE p=@b;
@pv2 := SELECT srcp.key,srcp.value FROM @srcp JOIN @a ON srcp.key=a.k;
END;
```

 **Note** Statements between BEGIN and END are the script of this view. The @pv2 :=... statement is similar to the RETURN statement in other programming languages. You can use the @pv2 :=... statement to assign a value to a variable of an implicit table that has the same name as the view.

- The matching rules for actual and formal view parameters are the same as those specified in a common weakly-typed language. Specifically, if a view parameter can be implicitly converted, it can match the defined parameter. For example, a value of the BIGINT type can match parameters of the DOUBLE type. For table variables, if the schema of table a can be inserted into table b, table a can be used to match the

table-type parameters that have the same schema as table b.

- You can explicitly declare the return type to make the code easier to read.

```
CREATE VIEW IF NOT EXISTS pv3(@a table (k STRING,v BIGINT), @b STRING)
RETURNS @ret TABLE (x STRING,y STRING)
AS
BEGIN
  @srcp := SELECT * FROM srcp WHERE p=@b;
  @ret := SELECT srcp.key,srcp.value FROM @srcp JOIN @a ON srcp.key=a.k;
end;
```

 **Note** RETURNS @ret TABLE (x STRING,y STRING) defines the following information:

- The return type, which indicates the type returned to the caller. The return type is specified by the TABLE (x STRING,y STRING) parameter. You can use this parameter to customize the table schema.
- The response parameter. The @ret parameter defines the name of the response parameter. Assigning a value to the response parameter is performed in the view script.

You can consider the view that contains no BEGIN/END or return variables as a simplified parameterized view.

1.6.11. CLONE TABLE

This topic describes how to use the CLONE TABLE statement. The CLONE TABLE statement is used to clone data from one table to another. This statement improves data migration efficiency.

Limits

- The schema of the destination table must be compatible with that of the source table.
- The CLONE TABLE statement can be executed for partitioned tables and non-partitioned tables. This statement cannot be executed for hash clustering tables.
- If a destination table is created before the CLONE TABLE statement is executed, data in a maximum of 10,000 partitions can be cloned at a time.
- If a destination table is not created before the CLONE TABLE statement is executed, the number of partitions from which you can clone data at a time is unlimited. This ensures atomicity.
- You can execute the CLONE TABLE statement for up to seven times in the same non-partitioned table or in the same partition of a partitioned table.
- You cannot execute the CLONE TABLE statement for projects across regions.

Syntax

```
CLONE TABLE <[src_project_name.]src_table_name> [PARTITION(spec), ...]
TO <[dest_project_name.]desc_table_name> [IF EXISTS (OVERWRITE | IGNORE)];
```

Description

The CLONE TABLE statement is used to clone data from the src_table_name table to the desc_table_name table.

Note After you clone data to the desc_table_name table, we recommend that you verify the data accuracy. For example, you can execute the SELECT COUNT statement to view the number of rows in the destination table or execute the DESC statement to view the table size.

Parameters

- src_table_name: the name of the source table.
- src_project_name: the name of the project to which the source table belongs. If this parameter is not specified, the name of the current project is used by default.
- desc_table_name: the name of the destination table.
 - If a destination table is not created, the table is created by using the CREATE TABLE LIKE statement when you execute the CLONE TABLE statement.
 - If a destination table is created and IF EXISTS OVERWRITE is specified, data in the partitions of the destination table is overwritten when you execute the CLONE TABLE statement.
 - If a destination table is created and IF EXISTS IGNORE is specified, the existing partitions in the table are skipped and the data in these partitions is not overwritten when you execute the CLONE TABLE statement.
- dest_project_name: the name of the project to which the destination table belongs. If this parameter is not specified, the current project name is used by default.

Examples

Assume that the partitioned table srcpart_copy and non-partitioned table src_copy are source tables. The two tables have the following metadata:

```
odps@ multi>READ srcpart_copy;
+-----+-----+-----+-----+
| key  | value | ds    | hr  |
+-----+-----+-----+-----+
| 1    | ok49  | 2008-04-09 | 11  |
| 1    | ok48  | 2008-04-08 | 12  |
+-----+-----+-----+-----+
odps@ multi>READ src_copy;
+-----+-----+
| key  | value |
+-----+-----+
| 1    | ok    |
+-----+-----+
```

- Clone all data from the non-partitioned table src_copy to the destination table src_clone.

```
CLONE TABLE src_copy TO src_clone;
```

Returned results:

```
ID = 2019102303024544g2540cdv2
OK
```

//After the data is cloned, query data in the destination table src_clone and check the data accuracy.

```
SELECT * FROM src_clone;
```

- Clone data from a specified partition of the partitioned table srcpart_copy to the destination table srcpart_clone.

```
CLONE TABLE srcpart_copy PARTITION(ds="2008-04-09", hr='11') TO srcpart_clone IF EXISTS OVERWRITE;
```

Returned results:

```
ID = 20191023030534986g4540cdv2
OK
```

//After the data is cloned, query data in the destination table srcpart_clone and check the data accuracy.

```
SELECT * FROM srcpart_clone;
```

- Clone all data from the partitioned table srcpart_copy to the destination table srcpart_clone and skip the data in the existing partitions of the destination table.

```
CLONE TABLE srcpart_copy TO srcpart_clone IF EXISTS IGNORE;
```

Returned results:

```
ID = 20191023030619196g5540cdv2
OK
```

//After the data is cloned, query data in the destination table srcpart_clone and check the data accuracy.

```
SELECT * FROM srcpart_clone;
```

- Clone all data from the partitioned table srcpart_copy to the destination table srcpart_clone2.

```
CLONE TABLE srcpart_copy TO srcpart_clone2;
```

Returned results:

```
ID = 20191023030825186g6540cdv2
OK
```

//After the data is cloned, query data in the destination table srcpart_clone2 and check the data accuracy.

```
SELECT * FROM srcpart_clone2;
```

1.6.12. Geographic functions

1.6.12.1. Usage notes

Before you use geographic functions, understand the following points:

All functions are published in the geospatial project of the DataWorks marketplace. These functions are prefixed with ST_. You can click a function to view and use it without the need to apply for permissions. To use a function, add a `geospatial.` project prefix to the beginning of the function name and commit SQL statements that contain this function with the following flags:

```
set odps.sql.hive.compatible=true;
set odps.sql.udf.java.retain.legacy=false;
set odps.isolation.session.enable=true;
```

1.6.12.2. Constructors

1.6.12.2.1. ST_AsBinary

This topic describes the Constructor function `ST_AsBinary` and provides an example for reference.

Function declaration:

```
ST_AsBinary(ST_Geometry)
```

Description: This function returns the well-known binary (WKB) expression of the input geometry.

Example:

```
SELECT ST_AsBinary(ST_Point(1, 2)) FROM onerow;
```

Returned results:

```
WKB representation of POINT (1 2)
```

1.6.12.2.2. ST_AsGeoJson

This topic describes the Constructor function `ST_AsGeoJson` and provides an example for reference.

Function declaration:

```
ST_AsGeoJson(geometry)
```

Description: This function returns the GeoJSON expression of the input geometry.

Example:

```
SELECT ST_AsGeoJson(ST_Point(1.0, 2.0)) from onerow;
```

Returned results:

```
{"type":"Point", "coordinates":[1.0, 2.0]}
```

1.6.12.2.3. ST_AsJson

This topic describes the Constructor function `ST_AsJson` and provides examples for reference.

Function declaration:

```
ST_AsJSON(ST_Geometry)
```

Description: This function returns the JSON expression of the input geometry.

Examples:

```
SELECT ST_AsJSON(ST_Point(1.0, 2.0)) from onerow;
```

Returned results:

```
{"x":1.0,"y":2.0}
```

```
SELECT ST_AsJSON(ST_SetSRID(ST_Point(1, 1), 4326)) from onerow;
```

Returned results:

```
{"x":1.0,"y":1.0,"spatialReference":{"wkid":4326}}
```

1.6.12.2.4. ST_AsShape

This topic describes the Constructor function `ST_AsShape` and provides an example for reference.

Function declaration:

```
ST_AsShape(ST_Geometry)
```

Description: This function returns the ESRI shape expression of the input geometry.

Example:

```
SELECT ST_AsShape(ST_Point(1,2)) FROM onerow;
```

Returned results:

```
Esri shape representation of POINT (1 2)
```

1.6.12.2.5. ST_AsText

This topic describes the Constructor function `ST_AsText` and provides an example for reference.

Function declaration:

```
ST_AsText(ST_Geometry)
```

Description: This function returns the well-known text (WKT) expression of the input geometry.

Example:

```
SELECT ST_AsText(ST_Point(1,2)) FROM onerow;
```

Returned results:

```
POINT (1 2)
```

1.6.12.2.6. ST_GeomCollection

This topic describes the `ST_GeomCollection` function of the Constructors function and provides an example for reference.

Function declaration:

```
ST_GeomCollection(wkt)
```

Description: This function constructs a multi-part geometry based on the well-known text (WKT) representation defined by the Open Geospatial Consortium (OGC).

 **Notice** The ST_GeomCollection function in MaxCompute supports only the multi-part geometry feature, not the collection feature.

Example:

```
SELECT ST_GeomCollection('multipoint ((1 0), (2 3))') FROM src LIMIT 1;
-- Construct a multipoint geometry.
ST_GeomCollection('POINT(1 1), LINESTRING(2 0,3 0)')
-- The collection feature is not supported.
```

1.6.12.2.7. ST_GeomFromGeoJson

This topic describes the Constructor function ST_GeomFromGeoJson and provides an example for reference.

Function declaration:

```
ST_GeomFromGeoJson(json)
```

Description: This function constructs a geometry based on the input GeoJSON expression.

Example:

```
SELECT ST_GeomFromGeoJson('{"type":"Point", "coordinates":[1.2, 2.4]}') FROM src LIMIT 1;
-- Construct a point.
SELECT ST_GeomFromGeoJson('{"type":"LineString", "coordinates":[[1,2], [3,4]]}') FROM src LIMIT 1;
-- Construct a linestring.
```

1.6.12.2.8. ST_GeomFromJSON

This topic describes the Constructor function ST_GeomFromJSON and provides an example for reference.

Function declaration:

```
ST_GeomFromJSON(json)
```

Description: This function constructs a geometry based on the input ESRI JSON expression.

Example:

```
SELECT ST_GeomFromJSON('{"x":0.0,"y":0.0}') FROM src LIMIT 1;
-- Construct a point.
```

1.6.12.2.9. ST_GeomFromShape

This topic describes the Constructor function ST_GeomFromShape and provides an example for reference.

Function declaration:

```
ST_GeomFromShape(shape)
```

Description: This function constructs a geometry based on the input ESRI shape expression.

Example:

```
SELECT ST_GeomFromShape(ST_AsShape(ST_Point(1, 2)));
-- Construct a point.
```

1.6.12.2.10. ST_GeomFromText

This topic describes the Constructor function `ST_GeomFromText` and provides an example for reference.

Function declaration:

```
ST_GeomFromText(wkt)
```

Description: This function constructs a geometry based on the input well-known text (WKT) representation defined by the Open Geospatial Consortium (OGC).

Example:

```
SELECT ST_GeomFromText('linestring (1 0, 2 3)') FROM src LIMIT 1;
-- Construct a linestring.
SELECT ST_GeomFromText('multipoint ((1 0), (2 3))') FROM src LIMIT 1;
-- Construct a multipoint geometry.
```

1.6.12.2.11. ST_GeomFromWKB

This topic describes the Constructor function `ST_GeomFromWKB` and provides an example for reference.

Function declaration:

```
ST_GeomFromWKB(wkb)
```

Description: This function constructs a geometry based on the input well-known binary (WKB) expression that complies with the Open Geospatial Consortium (OGC) standards.

Example:

```
SELECT ST_GeomFromWKB(ST_AsBinary(ST_GeomFromText('linestring (1 0, 2 3)'))) FROM src LIMIT 1;
-- Construct a linestring.
SELECT ST_GeomFromWKB(ST_AsBinary(ST_GeomFromText('multipoint ((1 0), (2 3))'))) FROM src LIMIT 1;
-- Construct a multipoint geometry.
```

1.6.12.2.12. ST_GeometryType

This topic describes the Constructor function `ST_GeometryType` and provides examples for reference.

Function declaration:

```
ST_GeometryType(geometry)
```

Description: This function returns the type of the input geometry.

Examples:

```
SELECT ST_GeometryType(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
ST_Point
```

```
SELECT ST_GeometryType(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
ST_LineString
```

```
SELECT ST_GeometryType(ST_Polygon(2,0, 2,3, 3,0)) FROM src LIMIT 1;
```

Returned results:

```
ST_Polygon
```

1.6.12.2.13. ST_LineString

This topic describes the Constructor function ST_LineString and provides an example for reference.

Function declaration:

```
ST_LineString(x, y, [x, y]*)  
ST_LineString('linestring( ... )')  
ST_LineString(array(x+), array(y+))  
ST_LineString(array(ST_Point(x,y+)))
```

Description: This function constructs a two-dimensional line.

Example:

```
SELECT ST_LineString(1, 1, 2, 2, 3, 3) from src LIMIT 1;  
SELECT ST_LineString('linestring(1 1, 2 2, 3 3)') from src LIMIT 1;  
SELECT ST_LineString(array(1,2,3), array (1,2,3)) from src LIMIT 1;  
SELECT ST_LineString(array(ST_Point(1, 1), ST_Point(2,2), ST_Point(3,3))) from src LIMIT 1;
```

1.6.12.2.14. ST_LineFromWKB

This topic describes the Constructor function ST_LineFromWKB and provides an example for reference.

Function declaration:

```
ST_LineFromWKB(wkb)
```

Description: This function constructs a two-dimensional line based on the input well-known binary (WKB) representation defined by the Open Geospatial Consortium (OGC).

Example:

```
SELECT ST_LineFromWKB(ST_AsBinary(ST_GeomFromText('linestring (1 0, 2 3)'))) FROM src LIMIT 1;  
-- Construct a two-dimensional line.
```

1.6.12.2.15. ST_MultiLineString

This topic describes the Constructor function ST_MultiLineString and provides an example for reference.

Function declaration:

```
ST_MultiLineString(array(x1, y1, x2, y2, ...), array(x1, y1, x2, y2, ...), ...)  
ST_MultiLineString('multilinestring( ...)')
```

Description: This function constructs a two-dimensional multilinestring.

Example:

```
SELECT ST_MultiLineString(array(1, 1, 2, 2), array(10, 10, 20, 20)) from src LIMIT 1;  
SELECT ST_MultiLineString('multilinestring ((1 1, 2 2), (10 10, 20 20))', 0) from src LIMIT 1;  
-- Construct a two-dimensional multilinestring.
```

1.6.12.2.16. ST_MLineFromWKB

This topic describes the Constructor function ST_MLineFromWKB and provides an example for reference.

Function declaration:

```
ST_MLineFromWKB(wkb)
```

Description: This function constructs a two-dimensional multilinestring based on the input well-known binary (WKB) expression that complies with the Open Geospatial Consortium (OGC) standards.

Example:

```
SELECT ST_MLineFromWKB(ST_AsBinary(ST_GeomFromText('multilinestring ((1 0, 2 3), (5 7, 7 5))'))) FROM src LIMIT  
1;  
-- Construct a two-dimensional multilinestring.
```

1.6.12.2.17. ST_MultiPoint

This topic describes the Constructor function ST_MultiPoint and provides an example for reference.

Function declaration:

```
ST_MultiPoint(x1, y1, x2, y2, x3, y3)  
ST_MultiPoint('multipoint( ...)')
```

Description: This function constructs a two-dimensional multipoint geometry.

Example:

```
SELECT ST_MultiPoint(1, 1, 2, 2, 3, 3) from src LIMIT 1;  
-- Construct a three-point geometry.  
SELECT ST_MultiPoint('MULTIPOINT ((10 40), (40 30))') from src LIMIT 1;  
-- Construct a two-point geometry.
```

1.6.12.2.18. ST_MPointFromWKB

This topic describes the Constructor function `ST_MPointFromWKB` and provides an example for reference.

Function declaration:

```
ST_MPointFromWKB(wkb)
```

Description: This function constructs a two-dimensional multipoint geometry based on the input well-known binary (WKB) representation defined by the Open Geospatial Consortium (OGC).

Example:

```
SELECT ST_MPointFromWKB(ST_AsBinary(ST_GeomFromText('multipoint ((1 0), (2 3))')) FROM src LIMIT 1;
-- Construct a two-dimensional multipoint geometry.
```

1.6.12.2.19. ST_MultiPolygon

This topic describes the Constructor function `ST_MultiPolygon` and provides an example for reference.

Function declaration:

```
ST_MultiPolygon(array(x1, y1, x2, y2, ...), array(x1, y1, x2, y2, ...), ... )
ST_MultiPolygon('multipolygon (...)' )
```

Description: This function constructs a two-dimensional multipolygon.

Example:

```
SELECT ST_MultiPolygon(array(1, 1, 1, 2, 2, 2, 1), array(3, 3, 3, 4, 4, 4, 3)) from src LIMIT 1;
SELECT ST_MultiPolygon('multipolygon (((0 0, 0 1, 1 0, 0 0)), ((2 2, 2 3, 3 2, 2 2)))') from src LIMIT 1;
-- Construct a two-dimensional multipolygon.
```

1.6.12.2.20. ST_MPolyFromWKB

This topic describes the Constructor function `ST_MPolyFromWKB` and provides an example for reference.

Function declaration:

```
ST_MPolyFromWKB(wkb)
```

Description: This function constructs a two-dimensional multipolygon based on the input well-known binary (WKB) expression that complies with the Open Geospatial Consortium (OGC) standards.

Example:

```
SELECT ST_MPolyFromWKB(ST_AsBinary(ST_GeomFromText('multipolygon (((0 0, 1 0, 0 1, 0 0)), ((2 2, 1 2, 2 1, 2 2)))')) FROM src LIMIT 1;
-- Construct a two-dimensional multipolygon.
```

1.6.12.2.21. ST_Point

This topic describes the Constructor function `ST_Point` and provides an example for reference.

Function declaration:

```
ST_Point(x, y)
ST_Point('point (x y)')
```

Description: This function constructs a two-dimensional point.

Example:

```
SELECT ST_Point(longitude, latitude) from src LIMIT 1;
SELECT ST_Point('point (0 0)') from src LIMIT 1;
-- Construct a two-dimensional point.
```

1.6.12.2.22. ST_PointFromWKB

This topic describes the Constructor function ST_PointFromWKB and provides an example for reference.

Function declaration:

```
ST_PointFromWKB(wkb)
```

Description: This function constructs a two-dimensional point based on the input well-known binary (WKB) expression that complies with the Open Geospatial Consortium (OGC) standards.

Example:

```
SELECT ST_PointFromWKB(ST_AsBinary(ST_GeomFromText('point (1 0)')))) FROM src LIMIT 1;
-- Construct a two-dimensional point.
```

1.6.12.2.23. ST_PointZ

This topic describes the Constructor function ST_PointZ and provides an example for reference.

Function declaration:

```
ST_PointZ(x, y, z)
```

Description: This function constructs a three-dimensional point.

Example:

```
SELECT ST_PointZ(longitude, latitude, elevation) from src LIMIT 1;
-- Construct a three-dimensional point.
```

1.6.12.2.24. ST_Polygon

This topic describes the Constructor function ST_Polygon and provides an example for reference.

Function declaration:

```
ST_Polygon(x, y, [x, y]*)
ST_Polygon('polygon( ... )')
```

Description: This function constructs a two-dimensional polygon.

Example:

```
SELECT ST_Polygon(1, 1, 1, 4, 4, 4, 1) from src LIMIT 1;
-- Construct a square.
SELECT ST_Polygon('polygon ((1 1, 4 1, 1 4))') from src LIMIT 1;
-- Construct a triangle.
```

1.6.12.2.25. ST_PolyFromWKB

This topic describes the Constructor function ST_PolyFromWKB and provides an example for reference.

Function declaration:

```
ST_PolyFromWKB(wkb)
```

Description: This function constructs a two-dimensional polygon based on the input well-known binary (WKB) representation defined by the Open Geospatial Consortium (OGC).

Example:

```
SELECT ST_PolyFromWKB(ST_AsBinary(ST_GeomFromText('polygon ((0 0, 10 0, 0 10, 0 0))')) FROM src LIMIT 1;
-- Construct a two-dimensional polygon.
```

1.6.12.2.26. ST_SetSRID

This topic describes the Constructor function ST_SetSRID and provides an example for reference.

Function declaration:

```
ST_SetSRID(<ST_Geometry>, SRID)
```

Description: This function sets the spatial reference system identifier (SRID) of the input geometry.

Example:

```
SELECT ST_SetSRID(ST_Point(1.5, 2.5), 4326) FROM src LIMIT 1;
-- Construct a point and set its SRID to 4326.
```

1.6.12.3. Accessors

1.6.12.3.1. ST_Area

This topic describes the Accessor function ST_Area and provides an example for reference.

Function declaration:

```
ST_Area(ST_Polygon)
```

Description: This function returns the areas of one or more polygons.

Example:

```
SELECT ST_Area(ST_Polygon(1,1, 1,4, 4,4, 4,1)) FROM src LIMIT 1;
```

Returned results:

9.0

1.6.12.3.2. ST_Centroid

This topic describes the Accessor function ST_Centroid and provides examples for reference.

Function declaration:

```
ST_Centroid(polygon)
```

Description: This function returns the centroid of the minimum bounding rectangle of the input polygon.

Examples:

```
SELECT ST_Centroid(ST_GeomFromText('polygon ((0 0, 3 6, 6 0, 0 0))')) FROM src LIMIT 1;
```

Returned results:

```
POINT(3 3)
```

```
SELECT ST_Centroid(ST_GeomFromText('polygon ((0 0, 0 8, 8 0, 0 0))')) FROM src LIMIT 1;
```

Returned results:

```
POINT(4 4)
```

1.6.12.3.3. ST_CoordDim

This topic describes the Accessor function ST_CoordDim and provides examples for reference.

Function declaration:

```
ST_CoordDim(geometry)
```

Description: This function returns the coordinate dimension of the input geometry.

Examples:

```
SELECT ST_CoordDim(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_CoordDim(ST_PointZ(1.5,2.5, 3)) FROM src LIMIT 1;
```

Returned results:

```
3
```

```
SELECT ST_CoordDim(ST_Point(1.5, 2.5, 3., 4.)) FROM src LIMIT 1;
```

Returned results:

```
4
```

1.6.12.3.4. ST_Dimension

This topic describes the Accessor function ST_Dimension and provides examples for reference.

Function declaration:

```
ST_Dimension(geometry)
```

Description: This function returns the spatial dimension of the input geometry.

Examples:

```
SELECT ST_Dimension(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
0
```

```
SELECT ST_Dimension(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
1
```

```
SELECT ST_Dimension(ST_Polygon(2,0, 2,3, 3,0)) FROM src LIMIT 1;
```

Returned results:

```
2
```

1.6.12.3.5. ST_Distance

This topic describes the Accessor function ST_Distance and provides an example for reference.

Function declaration:

```
ST_Distance(ST_Geometry1, ST_Geometry2)
```

Description: This function returns the distance between two geometries.

Example:

```
SELECT ST_Distance(ST_Point(0.0,0.0), ST_Point(3.0,4.0)) FROM src LIMIT 1;
```

Returned results:

5.0

1.6.12.3.6. ST_GeodesicLengthWGS84

This topic describes the Accessor function ST_GeodesicLengthWGS84 and provides examples for reference.

Function declaration:

```
ST_GeodesicLengthWGS84(line)
```

Description: This function returns the distance in meters on a spheroid based on World Geodetic System 1984 (WGS84). The geometry must be in WGS84. Otherwise, this function returns NULL.

Examples:

```
SELECT ST_GeodesicLengthWGS84(ST_SetSRID(ST_Linestring(0.0,0.0, 0.3,0.4), 4326)) FROM src LIMIT 1;
```

Returned results:

```
55km
```

```
SELECT ST_GeodesicLengthWGS84(ST_GeomFromText('MultiLineString((0.0 80.0, 0.3 80.4))', 4326)) FROM src LIMIT 1;
```

Returned results:

```
45km
```

1.6.12.3.7. ST_GeometryN

This topic describes the Accessor function ST_GeometryN and provides examples for reference.

Function declaration:

```
ST_GeometryN(ST_GeometryCollection, n)
```

Description: This function returns the n^{th} geometry in the input geometry collection. n starts from 1.

Examples:

```
SELECT ST_GeometryN(ST_GeomFromText('multipoint ((10 40), (40 30), (20 20), (30 10))'), 3) FROM src LIMIT 1;
```

Returned results:

```
ST_Point(20 20)
```

```
SELECT ST_GeometryN(ST_GeomFromText('multilinestring ((2 4, 10 10), (20 20, 7 8))'), 2) FROM src LIMIT 1;
```

Returned results:

```
ST_Linestring(20 20, 7 8)
```

1.6.12.3.8. ST_Is3D

This topic describes the Accessor function ST_Is3D and provides an example for reference.

Function declaration:

```
ST_Is3D(geometry)
```

Description: If the input geometry has Z coordinates, this function returns True. Otherwise, this function returns False.

Example:

```
SELECT ST_Is3D(ST_Polygon(1,1, 1,4, 4,4, 4,1)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_Is3D(ST_LineString(0.,0., 3.,4., 0.,4., 0.,0.)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_Is3D(ST_Point(3., 4.)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_Is3D(ST_PointZ(3., 4., 2)) FROM src LIMIT 1;
-- True is returned.
```

1.6.12.3.9. ST_IsClosed

This topic describes the Accessor function ST_IsClosed and provides an example for reference.

Function declaration:

```
ST_IsClosed(ST_[Multi]LineString)
```

Description: If the input linestring or linestrings are closed, this function returns True. Otherwise, it returns False.

Example:

```
SELECT ST_IsClosed(ST_LineString(0.,0., 3.,4., 0.,4., 0.,0.)) FROM src LIMIT 1;
-- True is returned.
SELECT ST_IsClosed(ST_LineString(0.,0., 3.,4.)) FROM src LIMIT 1;
-- False is returned.
```

1.6.12.3.10. ST_IsEmpty

This topic describes the Accessor function ST_IsEmpty and provides an example for reference.

Function declaration:

```
ST_IsEmpty(geometry)
```

Description: If the input geometry is empty, this function returns True. Otherwise, it returns False.

Example:

```
SELECT ST_IsEmpty(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_IsEmpty(ST_GeomFromText('point empty')) FROM src LIMIT 1;
-- True is returned.
```

1.6.12.3.11. ST_IsMeasured

This topic describes the Accessor function `ST_IsMeasured` and provides an example for reference.

Function declaration:

```
ST_IsMeasured(geometry)
```

Description: If the input `geometry` has `M` coordinates (measures), this function returns `True`. Otherwise, it returns `False`.

Example:

```
SELECT ST_IsMeasured(ST_Polygon(1,1, 1,4, 4,4, 4,1)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_IsMeasured(ST_LineString(0.,0., 3.,4., 0.,4., 0.,0.)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_IsMeasured(ST_Point(3., 4.)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_IsMeasured(ST_PointM(3., 4., 2)) FROM src LIMIT 1;
-- True is returned.
```

1.6.12.3.12. ST_IsSimple

This topic describes the Accessor function `ST_IsSimple` and provides an example for reference.

Function declaration:

```
ST_IsSimple(geometry)
```

Description: If the input `geometry` is a simple object, this function returns `True`. Otherwise, it returns `False`.

Example:

```
SELECT ST_IsSimple(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
-- True is returned.
SELECT ST_IsSimple(ST_LineString(0.,0., 1.,1., 0.,1., 1.,0.)) FROM src LIMIT 1;
-- False is returned.
```

1.6.12.3.13. ST_IsRing

This topic describes the Accessor function `ST_IsRing` and provides an example for reference.

Function declaration:

```
ST_IsRing(ST_LineString)
```

Description: This function returns `True` if the line string is a simple object or closed. Otherwise, it returns `False`.

Example:

```
SELECT ST_IsRing(ST_LineString(0,0, 3,4, 0,4, 0,0)) FROM src LIMIT 1;
-- True is returned.
SELECT ST_IsRing(ST_LineString(0,0, 1,1, 1,2, 2,1, 1,1, 0,0)) FROM src LIMIT 1;
-- False is returned.
SELECT ST_IsRing(ST_LineString(0,0, 3,4)) FROM src LIMIT 1;
-- False is returned.
```

1.6.12.3.14. ST_Length

This topic describes the Accessor function ST_Length and provides an example for reference.

Function declaration:

```
ST_Length(line)
```

Description: This function returns the length of the input line segment.

Example:

```
SELECT ST_Length(ST_Line(0,0,0,0, 3,0,4,0)) FROM src LIMIT 1;
```

Returned results:

```
5.0
```

1.6.12.3.15. ST_M

This topic describes the Accessor function ST_M and provides an example for reference.

Function declaration:

```
ST_M(geometry)
```

Description: This function returns the M coordinate of the input geometry.

Example:

```
SELECT ST_M(ST_PointM(3., 4., 2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

1.6.12.3.16. ST_MaxM

This topic describes the Accessor function ST_MaxM and provides examples for reference.

Function declaration:

```
ST_MaxM(geometry)
```

Description: This function returns the maximum M coordinate of the input geometry.

Examples:

```
SELECT ST_MaxM(ST_PointM(1.5, 2.5, 2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_MaxM(ST_LineString('linestring m (1.5 2.5 2, 3.0 2.2 1)')) FROM src LIMIT 1;
```

Returned results:

```
1
```

1.6.12.3.17. ST_MinM

This topic describes the Accessor function ST_MinM and provides examples for reference.

Function declaration:

```
ST_MinM(geometry)
```

Description: This function returns the minimum M coordinate of the input geometry.

Examples:

```
SELECT ST_MinM(ST_PointM(1.5, 2.5, 2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_MinM(ST_LineString('linestring m (1.5 2.5 2, 3.0 2.2 1)')) FROM src LIMIT 1;
```

Returned results:

```
1
```

1.6.12.3.18. ST_X

This topic describes the Accessor function ST_X and provides an example for reference.

Function declaration:

```
ST_X(point)
```

Description: This function returns the X coordinate of the input point.

Example:

```
SELECT ST_X(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
1.5
```

1.6.12.3.19. ST_Y

This topic describes the Accessor function ST_Y and provides an example for reference.

Function declaration:

```
ST_Y(point)
```

Description: This function returns the Y coordinate of the input point.

Example:

```
SELECT ST_Y(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
2.5
```

1.6.12.3.20. ST_Z

This topic describes the Accessor function ST_Z and provides an example for reference.

Function declaration:

```
ST_Z(point)
```

Description: This function returns the Z coordinate of the input point.

Example:

```
SELECT ST_Z(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
1.5
```

1.6.12.3.21. ST_MaxX

This topic describes the Accessor function ST_MaxX and provides examples for reference.

Function declaration:

```
ST_MaxX(geometry)
```

Description: This function returns the maximum X coordinate of the input geometry.

Examples:

```
SELECT ST_MaxX(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
1.5
```

```
SELECT ST_MaxX(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
3.0
```

1.6.12.3.22. ST_MaxY

This topic describes the Accessor function ST_MaxY and provides examples for reference.

Function declaration:

```
ST_MaxY(geometry)
```

Description: This function returns the maximum Y coordinate of the input geometry.

Examples:

```
SELECT ST_MaxY(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
2.5
```

```
SELECT ST_MaxY(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
2.5
```

1.6.12.3.23. ST_MaxZ

This topic describes the Accessor function ST_MaxZ and provides examples for reference.

Function declaration:

```
ST_MaxZ(geometry)
```

Description: This function returns the maximum Z coordinate of the input geometry.

Examples:

```
SELECT ST_MaxZ(ST_PointZ(1.5, 2.5, 2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_MaxZ(ST_LineString('linestring z (1.5 2.5 2, 3.0 2.2 1)')) FROM src LIMIT 1;
```

Returned results:

```
1
```

1.6.12.3.24. ST_MinX

This topic describes the Accessor function ST_MinX and provides examples for reference.

Function declaration:

```
ST_MinX(geometry)
```

Description: This function returns the minimum X coordinate of the input geometry.

Examples:

```
SELECT ST_MinX(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
1.5
```

```
SELECT ST_MinX(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
3.0
```

1.6.12.3.25. ST_MinY

This topic describes the Accessor function ST_MinY and provides examples for reference.

Function declaration:

```
ST_MinY(geometry)
```

Description: This function returns the minimum Y coordinate of the input geometry.

Examples:

```
SELECT ST_MinY(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
2.5
```

```
SELECT ST_MinY(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
2.2
```

1.6.12.3.26. ST_MinZ

This topic describes the Accessor function ST_MinZ and provides examples for reference.

Function declaration:

```
ST_MinZ(geometry)
```

Description: This function returns the minimum Z coordinate of the input geometry.

Examples:

```
SELECT ST_MinZ(ST_PointZ(1.5, 2.5, 2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_MinZ(ST_LineString('linestring z (1.5 2.5 2, 3.0 2.2 1)')) FROM src LIMIT 1;
```

Returned results:

```
1
```

1.6.12.3.27. ST_NumGeometries

This topic describes the Accessor function ST_NumGeometries and provides examples for reference.

Function declaration:

```
ST_NumGeometries(ST_GeometryCollection)
```

Description: This function returns the number of geometries in the input geometry collection.

Examples:

```
SELECT ST_NumGeometries(ST_GeomFromText('multipoint ((10 40), (40 30), (20 20), (30 10))')) FROM src LIMIT 1;
```

Returned results:

```
4
```

```
SELECT ST_NumGeometries(ST_GeomFromText('multilinestring ((2 4, 10 10), (20 20, 7 8))')) FROM src LIMIT 1;
```

Returned results:

2

1.6.12.3.28. ST_NumInteriorRing

This topic describes the Accessor function ST_NumInteriorRing and provides examples for reference.

Function declaration:

```
ST_NumInteriorRing(ST_Polygon)
```

Description: This function returns the number of interior rings of the input polygon.

Examples:

```
SELECT ST_NumInteriorRing(ST_Polygon(1,1, 1,4, 4,1)) FROM src LIMIT 1;
```

Returned results:

```
0
```

```
SELECT ST_NumInteriorRing(ST_Polygon('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))')) FROM src LIMIT 1;
```

Returned results:

```
1
```

1.6.12.3.29. ST_NumPoints

This topic describes the Accessor function ST_NumPoints and provides examples for reference.

Function declaration:

```
ST_NumPoints(geometry)
```

Description: This function returns the number of points in the input geometry.

Examples:

```
SELECT ST_NumPoints(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
1
```

```
SELECT ST_NumPoints(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
2
```

```
SELECT ST_NumPoints(ST_GeomFromText('polygon ((0 0, 10 0, 0 10, 0 0))')) FROM src LIMIT 1;
```

Returned results:

```
4
```

1.6.12.3.30. ST_PointN

This topic describes the Accessor function `ST_PointN` and provides an example for reference.

Function declaration:

```
ST_PointN(ST_Geometry, n)
```

Description: This function returns the n^{th} point of one or more linestrings.

Example:

```
SELECT ST_PointN(ST_LineString(1.5,2.5, 3.0,2.2), 2) FROM src LIMIT 1;
```

Returned results:

```
POINT(3.0 2.2)
```

1.6.12.3.31. ST_StartPoint

This topic describes the Accessor function `ST_StartPoint` and provides an example for reference.

Function declaration:

```
ST_StartPoint(geometry)
```

Description: This function returns the first point of the input linestring.

Example:

```
SELECT ST_StartPoint(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
POINT(1.5 2.5)
```

1.6.12.3.32. ST_EndPoint

This topic describes the Accessor function `ST_EndPoint` and provides an example for reference.

Function declaration:

```
ST_EndPoint(geometry)
```

Description: This function returns the last point of the input linestring.

Example:

```
SELECT ST_EndPoint(ST_LineString(1.5,2.5, 3.0,2.2)) FROM src LIMIT 1;
```

Returned results:

```
POINT(3.0 2.0)
```

1.6.12.3.33. ST_SRID

This topic describes the Accessor function ST_SRID and provides an example for reference.

Function declaration:

```
ST_SRID(ST_Geometry)
```

Description: This function returns the spatial reference system identifier (SRID) of the input geometry.

Example:

```
SELECT ST_SRID(ST_Point(1.5, 2.5)) FROM src LIMIT 1;
```

Returned results:

```
returns SRID 0
```

1.6.12.4. Operations

1.6.12.4.1. ST_Aggr_ConvexHull

This topic describes the Operation function ST_Aggr_ConvexHull and provides an example for reference.

Function declaration:

```
ST_Aggr_ConvexHull(ST_Geometry)
```

Description: This function returns a convex hull for input geometries by using aggregate transformation.

Example:

```
SELECT ST_Aggr_ConvexHull(geometry) FROM source;  
-- Return the convex hull of input geometries from the data source by using aggregate transformation.
```

1.6.12.4.2. ST_Aggr_Intersection

This topic describes the Operation function ST_Aggr_Intersection and provides an example for reference.

Function declaration:

```
ST_Aggr_Intersection(ST_Geometry)
```

Description: This function returns the intersection of input geometries by using aggregate transformation.

Example:

```
SELECT ST_Aggr_Intersection(geometry) FROM source;  
-- Return the intersection of input geometries from the data source by using aggregate transformation.
```

1.6.12.4.3. ST_Aggr_Union

This topic describes the Operation function ST_Aggr_Union and provides an example for reference.

Function declaration:

```
ST_Aggr_Union(ST_Geometry)
```

Description: This function returns a union of input geometries by using aggregate transformation.

Example:

```
SELECT ST_Aggr_Union(geometry) FROM source;  
-- Return the union of input geometries from the data source by using aggregate transformation.
```

1.6.12.4.4. ST_Bin

This topic describes the Operation function ST_Bin.

Function declaration:

```
ST_Bin(placeholder)
```

Description: This function returns the bin ID of the input point.

1.6.12.4.5. ST_BinEnvelope

This topic describes the Operation function ST_BinEnvelope.

Function declaration:

```
ST_BinEnvelope(binsize, point)
```

Description: This function returns the binary envelope for the input point.

Function declaration:

```
ST_BinEnvelope(binsize, binid)
```

Description: This function returns the binary envelope for the input bin ID.

1.6.12.4.6. ST_Boundary

This topic describes the Operation function ST_Boundary and provides examples for reference.

Function declaration:

```
ST_Boundary(ST_Geometry)
```

Description: This function returns the boundary of the input geometry ST_Geometry.

Examples:

```
SELECT ST_Boundary(ST_LineString(0,1, 1,0)) FROM src LIMIT 1;
```

Returned results:

```
MULTIPOINT((1 0),(0 1))
```

```
SELECT ST_Boundary(ST_Polygon(1,1, 4,1, 1,4)) FROM src LIMIT 1;
```

Returned results:

```
LINestring(1 1, 4 1, 1 4, 1 1)
```

1.6.12.4.7. ST_Buffer

This topic describes the Operation function ST_Buffer.

Function declaration:

```
ST_Buffer(geometry, distance)
```

Description: This function returns a geometry that indicates all points whose distance from this geometry to the input geometry is less than or equal to the value of the distance parameter.

1.6.12.4.8. ST_ConvexHull

This topic describes the Operation function ST_ConvexHull and provides an example for reference.

Function declaration:

```
ST_ConvexHull(ST_Geometry, ST_Geometry, ...)
```

Description: This function returns the geometric object ST_Geometry as the convex hull of the specified geometry.

Example:

```
SELECT ST_AsText(ST_ConvexHull(ST_Point(0, 0), ST_Point(0, 1), ST_Point(1, 1))) FROM onerow;
```

Returned results:

```
MULTIPOLYGON (((0 0, 1 1, 0 1, 0 0)))
```

1.6.12.4.9. ST_Difference

This topic describes the Operation function ST_Difference and provides examples for reference.

Function declaration:

```
ST_Difference(ST_Geometry1, ST_Geometry2)
```

Description: This function returns a geometry that indicates the difference between ST_Geometry1 and ST_Geometry2.

Examples:

```
SELECT ST_AsText(ST_Difference(ST_MultiPoint(1, 1, 1.5, 1.5, 2, 2), ST_Point(1.5, 1.5))) FROM onerow;
```

Returned result:

```
MULTIPOINT (1 1, 2 2)
```

```
SELECT ST_AsText(ST_Difference(ST_Polygon(0, 0, 0, 10, 10, 10, 10, 0), ST_Polygon(0, 0, 0, 5, 5, 5, 0))) from onerow;
```

Returned result:

```
MULTIPOLYGON (((10 0, 10 10, 0 10, 0 5, 5 5, 5 0, 10 0)))
```

1.6.12.4.10. ST_Envelope

This topic describes the Operation function ST_Envelope and provides examples for reference.

Function declaration:

```
ST_Envelope(ST_Geometry)
```

Description: This function returns the envelope of the input geometry. If the specified geometry is a point, a horizontal line, or a vertical line, this function returns the common difference or an empty envelope.

Examples:

```
SELECT ST_Envelope(ST_LineString(0,0, 2,2)) from src LIMIT 1;
```

Returned result:

```
POLYGON ((0 0, 2 0, 2 2, 0 2, 0 0))
```

```
SELECT ST_Envelope(ST_Polygon(2,0, 2,3, 3,0)) from src LIMIT 1;
```

Returned result:

```
POLYGON ((2 0, 3 0, 3 3, 2 3, 2 0))
```

1.6.12.4.11. ST_ExteriorRing

This topic describes the Operation function ST_ExteriorRing and provides examples for reference.

Function declaration:

```
ST_ExteriorRing(polygon)
```

Description: This function returns the exterior ring of a polygon as a linestring.

Examples:

```
SELECT ST_ExteriorRing(ST_Polygon(1,1, 1,4, 4,1)) FROM src LIMIT 1;
```

Returned results:

```
LINestring(1 1, 4 1, 1 4, 1 1)
```

```
SELECT ST_ExteriorRing(ST_Polygon('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))')) FROM src LIMIT 1;
```

Returned results:

```
LINestring (8 0, 0 8, 0 0, 8 0)
```

1.6.12.4.12. ST_InteriorRingN

This topic describes the Operation function ST_InteriorRingN and provides an example for reference.

Function declaration:

```
ST_InteriorRingN(ST_Polygon, n)
```

Description: This function returns the n^{th} interior ring of a polygon as a linestring.

Example:

```
SELECT ST_InteriorRingN(ST_Polygon('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1 1))'), 1) FROM src LIMIT 1;
```

Returned results:

```
LINestring (1 1, 5 1, 1 5, 1 1)
```

1.6.12.4.13. ST_Intersection

This topic describes the Operation function ST_Intersection and provides examples for reference.

Function declaration:

```
ST_Intersection(ST_Geometry1, ST_Geometry2)
```

Description: This function returns a geometry that indicates the intersection of the input geometries. If the input geometries intersect in a lower dimension, ST_Intersection may drop lower-dimension intersections or return a closed linestring.

Examples:

```
SELECT ST_AsText(ST_Intersection(ST_Point(1,1), ST_Point(1,1))) FROM onerow;
```

Returned results:

```
POINT (1 1)
```

```
SELECT ST_AsText(ST_Intersection(ST_GeomFromText('linestring(0 2, 0 0, 2 0)'), ST_GeomFromText('linestring(0 3, 0 1, 1 0, 3 0)'))) FROM onerow;
```

Returned results:

```
MULTILINESTRING ((1 0, 2 0), (0 2, 0 1))
```

```
SELECT ST_AsText(ST_Intersection(ST_LineString(0,2, 2,3), ST_Polygon(1,1, 4,1, 4,4, 1,4))) FROM onerow;
```

Returned results:

```
MULTILINESTRING ((1 2.5, 2 3))
```

```
SELECT ST_AsText(ST_Intersection(ST_Polygon(2,0, 2,3, 3,0), ST_Polygon(1,1, 4,1, 4,4, 1,4))) FROM onerow;
```

Returned result:

```
MULTIPOLYGON (((2.67 1, 2 3, 2 1, 2.67 1)))
```

```
SELECT ST_AsText(ST_Intersection(ST_Polygon(2,0, 3,1, 2,1), ST_Polygon(1,1, 4,1, 4,4, 1,4))) FROM onerow;
```

Returned results:

```
MULTIPOLYGON EMPTY or LINESTRING (2 1, 3 1, 2 1)
```

1.6.12.4.14. ST_SymmetricDiff

This topic describes the ST_SymmetricDiff function and provides examples of using this function.

Function declaration:

```
ST_SymmetricDiff(ST_Geometry1, ST_Geometry2)
```

Description: This function returns a geometry that consists of the symmetric differences of the input geometries.

Examples:

```
SELECT ST_AsText(ST_SymmetricDiff(ST_LineString('linestring(0 2, 2 2)'), ST_LineString('linestring(1 2, 3 2)'))) FROM onerow;
```

Returned result:

```
MULTILINESTRING((0 2, 1 2), (2 2, 3 2))
```

```
SELECT ST_AsText(ST_SymmetricDiff(ST_SymmetricDiff(ST_Polygon('polygon((0 0, 2 0, 2 2, 0 2, 0 0))'), ST_Polygon('polygon((1 1, 3 1, 3 3, 1 3, 1 1)')))) from onerow;
```

Returned result:

```
MULTIPOLYGON (((0 0, 2 0, 2 1, 1 1, 1 2, 0 2, 0 0)), ((3 1, 3 3, 1 3, 1 2, 2 2, 2 1, 3 1)))
```

1.6.12.4.15. ST_Union

This topic describes the ST_Union function and provides an example of using this function.

Function declaration:

```
ST_Union(ST_Geometry, ST_Geometry, ...)
```

Description: This function returns a geometry that is the union of the input geometries.

Example:

```
SELECT ST_AsText(ST_Union(ST_Polygon(1, 1, 1, 4, 4, 4, 4, 1), ST_Polygon(4, 1, 4, 4, 4, 8, 8, 1))) FROM onerow;
```

Returned result:

```
MULTIPOLYGON (((4 1, 8 1, 4 8, 4 4, 1 4, 1 1, 4 1)))
```

1.6.12.5. Relationship tests

1.6.12.5.1. ST_Contains

This topic describes the ST_Contains function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Contains(geometry1, geometry2)
```

Description: If geometry1 contains geometry2, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Contains(st_polygon(1,1, 1,4, 4,4, 4,1), st_point(2, 3) from src LIMIT 1;
-- true is returned.
SELECT ST_Contains(st_polygon(1,1, 1,4, 4,4, 4,1), st_point(8, 8) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.2. ST_Crosses

This topic describes the ST_Crosses function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Crosses(geometry1, geometry2)
```

Description: If geometry1 crosses geometry2, this function returns true. Otherwise, this function returns false.

 **Note** Crossing indicates that some points in the two geometries are the same.

Example:

```
SELECT ST_Crosses(st_linestring(0,0,1,1), st_linestring(1,0,0,1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Crosses(st_linestring(2,0,2,3), st_polygon(1,1,1,4,4,4,4,1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Crosses(st_linestring(0,2,0,1), ST_linestring(2,0,1,0)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.3. ST_Disjoint

This topic describes the ST_Disjoint function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Disjoint(geometry1, geometry2)
```

Description: If geometry1 and geometry2 do not intersect, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Disjoint(ST_LineString(0,0,0,1), ST_LineString(1,1,1,0)) from src LIMIT 1;
-- true is returned.
SELECT ST_Disjoint(ST_LineString(0,0,1,1), ST_LineString(1,0,0,1)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.4. ST_EnvIntersects

This topic describes the ST_EnvIntersects function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_EnvIntersects(ST_Geometry1, ST_Geometry2)
```

Description: If the envelopes of ST_Geometry1 and ST_Geometry2 intersect, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_EnvIntersects(ST_LineString(0,0,1,1), ST_LineString(1,3,2,2)) from src LIMIT 1;
-- false is returned.
SELECT ST_EnvIntersects(ST_LineString(0,0,2,2), ST_LineString(1,0,3,2)) from src LIMIT 1;
-- true is returned.
```

1.6.12.5.5. ST_Equals

This topic describes the ST_Equals function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Equals(geometry1, geometry2)
```

Description: If geometry1 equals geometry2, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Equals(st_linestring(0,0, 1,1), st_linestring(1,1, 0,0)) from src LIMIT 1;
-- true is returned.
SELECT ST_Equals(st_linestring(0,0, 1,1), st_linestring(1,0, 0,1)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.6. ST_Intersects

This topic describes the ST_Intersects function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Intersects(geometry1, geometry2)
```

Description: If geometry1 and geometry2 intersect, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Intersects(st_linestring(0,0, 1,1), st_linestring(1,1, 0,0)) from src LIMIT 1;
-- true is returned.
SELECT ST_Intersects(st_linestring(0,0, 1,1), st_linestring(1,0, 0,1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Intersects(ST_LineString(2,0, 2,3), ST_Polygon(1,1, 4,1, 4,4, 1,4)) from src LIMIT 1;
-- true is returned.
SELECT ST_Intersects(ST_LineString(8,7, 7,8), ST_Polygon(1,1, 4,1, 4,4, 1,4)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.7. ST_Overlaps

This topic describes the ST_Overlaps function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Overlaps(geometry1, geometry2)
```

Description: If geometry1 and geometry2 overlap, this function returns true. Otherwise, this function returns false. Overlapping excludes the tangency of the geometries.

Example:

```
SELECT ST_Overlaps(st_polygon(2,0, 2,3, 3,0), st_polygon(1,1, 1,4, 4,4, 4,1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Overlaps(st_polygon(2,0, 2,1, 3,1), ST_Polygon(1,1, 1,4, 4,4, 4,1)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.8. ST_Relate

This topic describes the ST_Relate function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Relate(geometry1, geometry2)
```

Description: If geometry1 has the specified Dimensionally Extended nine-Intersection Model (DE-9IM) relationship with geometry2, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Relate(st_polygon(2,0, 2,1, 3,1), ST_Polygon(1,1, 1,4, 4,4, 4,1), '****T****') from src LIMIT 1;
-- true is returned.
SELECT ST_Relate(st_polygon(2,0, 2,1, 3,1), ST_Polygon(1,1, 1,4, 4,4, 4,1), 'T*****') from src LIMIT 1;
-- false is returned.
SELECT ST_Relate(st_linestring(0,0, 3,3), ST_linestring(1,1, 4,4), 'T*****') from src LIMIT 1;
-- true is returned.
SELECT ST_Relate(st_linestring(0,0, 3,3), ST_linestring(1,1, 4,4), '****T****') from src LIMIT 1;
-- false is returned.
```

1.6.12.5.9. ST_Touches

This topic describes the ST_Touches function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Touches(geometry1, geometry2)
```

Description: If geometry1 and geometry2 spatially touch and have no similar interior points, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Touches(st_point(1, 2), st_polygon(1, 1, 1, 4, 4, 4, 1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Touches(st_point(8, 8), st_polygon(1, 1, 1, 4, 4, 4, 1)) from src LIMIT 1;
-- false is returned.
```

1.6.12.5.10. ST_Within

This topic describes the ST_Within function and provides an example of using this function.

Function declaration:

```
BOOLEAN ST_Within(geometry1, geometry2)
```

Description: If geometry1 is within geometry2, this function returns true. Otherwise, this function returns false.

Example:

```
SELECT ST_Within(st_point(2, 3), st_polygon(1,1, 1,4, 4,4, 4,1)) from src LIMIT 1;
-- true is returned.
SELECT ST_Within(st_point(8, 8), st_polygon(1,1, 1,4, 4,4, 4,1)) from src LIMIT 1;
-- false is returned.
```

1.6.12.6. Geohash index functions

1.6.12.6.1. ST_GeoHash

This topic describes the ST_GeoHash function and provides an example of using this function.

Function declaration:

```
string ST_GeoHash(st_geometry geometry, integer precision=full_precision)  
string ST_GeoHash(double longitude, double latitude, integer precision=full_precision)
```

Description: This function returns the unique Geohash string of the specified point. This function uses a function with the ST_ prefix or the specified longitudes and latitudes as input parameters. If the precision parameter is not specified, the maximum precision is used.

Example:

```
SELECT ST_GeoHash(ST_Point(-102.849854, 36.451113), 8);  
SELECT ST_GeoHash(ST_GeomFromText('POINT(-102.849854 36.451113)'));  
SELECT ST_GeoHash(-102.849854, 36.451113, 10);
```

1.6.12.6.2. ST_PointFromGeoHash

This topic describes the ST_PointFromGeoHash function and provides an example of using this function.

Function declaration:

```
st_geometry ST_PointFromGeoHash(string geohash, integer precision=full_precision)
```

Description: This function returns a point based on the input Geohash value. If the precision parameter is not specified, the maximum precision is used.

Example:

```
SELECT ST_AsText(ST_PointFromGeoHash('9wqz7eep0eyq'));  
SELECT ST_AsText(ST_PointFromGeoHash('9wqz7eep0eyq', 4));
```

1.6.12.6.3. ST_EnvelopeFromGeoHash

This topic describes the ST_EnvelopeFromGeoHash function and provides an example of using this function.

Function declaration:

```
st_geometry ST_EnvelopeFromGeoHash(string geohash, integer precision=full_precision)
```

Description: This function returns the envelope of the specified precision based on the input Geohash value. If the precision parameter is not specified, the maximum precision is used.

Example:

```
SELECT ST_AsText(ST_EnvelopeFromGeoHash('9wqz7eep0eyq', 8));  
SELECT ST_AsText(ST_EnvelopeFromGeoHash('9wqz7eep0eyq'));
```

1.6.12.6.4. ST_GeoHashNeighbours

This topic describes the ST_GeoHashNeighbours function and provides an example of using this function.

Function declaration:

```
list_of_string ST_GeoHashNeighbours(double longitude, double latitude, integer precision)
```

Description: This function is a user-defined table-valued function (UDTF) that generates nine data records. This function returns nine Geohash strings of the current point and its eight neighboring points based on the input longitude, latitude, and precision. These parameters must be specified.

Example:

```
SELECT ST_GeoHashNeighbours(-102.849854, 36.451113, 10);
```

1.6.12.7. S2 mesh functions

1.6.12.7.1. ST_S2CellIdsFromGeom

This topic describes the ST_S2CellIdsFromGeom function and provides an example of using this function.

Function declaration:

```
list_of_string ST_S2CellIdsFromGeom(st_geometry geometry, integer level)
```

Description: This function overwrites the input geometry by using S2 cells at the specified level. Then, it returns the IDs of all S2 cells.

Example:

```
SELECT ST_S2CellIdsFromGeom(ST_Point(-102.849854, 36.451113), 4);  
SELECT ST_S2CellIdsFromGeom(ST_LineString('LINESTRING(-71.160281 42.258729,-71.160837 42.259113,-71.161144 42.25932)'), 17) as cellid;
```

1.6.12.7.2. ST_S2CellIdsFromText

This topic describes the ST_S2CellIdsFromText function and provides an example of using this function.

Function declaration:

```
list_of_string ST_S2CellIdsFromText(string wkt, integer level)
```

Description: This function overwrites the well-known text (WKT) representation of the input geometry by using S2 cells at the specified level. Then, it returns the IDs of all S2 cells.

Example:

```
SELECT ST_S2CellIdsFromText(ST_GeomFromText('POINT(-102.849854 36.451113)'), 4);  
SELECT ST_S2CellIdsFromText('LINESTRING(-71.160281 42.258729,-71.160837 42.259113,-71.161144 42.25932)', 17) as cellid;
```

1.6.12.7.3. ST_S2CellCenterPoint

This topic describes the ST_S2CellCenterPoint function and provides an example of using this function.

Function declaration:

```
st_point ST_S2CellCenterPoint(string cellId)
```

Description: This function calculates the center point of the cell specified by the cellId parameter in the input S2 cell.

Example:

```
SELECT ST_S2CellCenterPoint('549015');  
SELECT ST_AsText(ST_S2CellCenterPoint('89e37f091'));
```

1.6.12.7.4. ST_S2CellNeighbours

This topic describes the ST_S2CellNeighbours function and provides an example of using this function.

Function declaration:

```
list_of_string ST_S2CellNeighbours(string cellId, integer level)
```

Description: This function calculates the neighboring S2 cells of the cell specified by the cellId parameter at the specified level. Then, it returns the IDs of all neighboring S2 cells.

Example:

```
SELECT ST_S2CellNeighbours('549015', 10);  
SELECT ST_S2CellNeighbours('89e37f091', 16) as neighbour;
```

1.6.12.8. Geodesic functions

1.6.12.8.1. ST_AreaWGS84

This topic describes the ST_AreaWGS84 function and provides an example of using this function.

Function declaration:

```
double ST_AreaWGS84(st_geometry geometry)
```

Description: This function returns the approximate geodesic area of the input geometry based on World Geodetic System 1984 (WGS84). This function converts the coordinates of the input geometry from EPSG:4326 to EPSG:3857. Then, it calculates the plane area in square meters.

Example:

```
SELECT ST_AreaWGS84(ST_GeomFromText('POLYGON((743238 2967416,743238 2967450, 743265 2967450,743265.6 25 2967416,743238 2967416))'));
```

1.6.12.8.2. ST_DistanceWGS84

This topic describes the ST_DistanceWGS84 function and provides an example of using this function.

Function declaration:

```
double ST_DistanceWGS84(st_geometry geometry1, st_geometry geometry2)
```

Description: This function returns the approximate geodesic distance of the input geometry based on World Geodetic System 1984 (WGS84). This function converts the coordinates of the input geometry from EPSG:4326 to EPSG:3857. Then, it calculates the plane distance in meters.

Example:

```
SELECT ST_DistanceWGS84(ST_GeomFromText('POINT(-72.1235 42.3521)'), ST_GeomFromText('LINESTRING(-72.1260 42.45, -72.123 42.1546)'));
```

1.6.12.8.3. ST_BufferWGS84

This topic describes the ST_BufferWGS84 function and provides an example of using this function.

Function declaration:

```
st_geometry ST_BufferWGS84(st_geometry geometry, double radius)
```

Description: This function returns the approximate geodesic buffer of the input geometry based on World Geodetic System 1984 (WGS84). This function converts the coordinates of the input geometry from EPSG:4326 to EPSG:3857. Then, it calculates the plane buffer and converts the coordinates back to EPSG:4326.

Example:

```
SELECT ST_AsText(ST_BufferWGS84(ST_GeomFromText('POINT(-72.1235 42.3521)'), 10));
```

1.6.12.8.4. ST_GeodesicDistance

This topic describes the ST_GeodesicDistance function and provides an example of using this function.

Function declaration:

```
double ST_GeodesicDistance(double lon1, double lat1, double lon2, double lat2, string method = VINCENTY)
double ST_GeodesicDistance(st_geometry geo1, st_geometry geo2, string method = VINCENTY)
```

Description: This function calculates the geodesic distance between two points by using the specified method. The supported methods are Vincenty, LawOfCosines, and Haversine. The default value of the method parameter is VINCENTY. The return value is in radians.

Example:

```
SELECT ST_GeodesicDistance(ST_GeomFromText('POINT(152.352298 -24.875975)'), ST_GeomFromText('POINT(151.960336 -24.993289)'), 'LawOfCosines');
```

1.6.12.8.5. ST_Distance_Sphere

This topic describes the ST_Distance_Sphere function and provides an example of using this function.

Function declaration:

```
double ST_Distance_Sphere(st_point geo1, st_point2 geo2)
double ST_Distance_Sphere(double lng1, double lat1, double lng2, double lat2)
```

Description: This function uses the algorithm provided by AMAP to calculate the approximate geodesic distance between the two input points. This function uses ST_Point or the specified longitudes and latitudes as input parameters.

Example:

```
SELECT ST_Distance_Sphere(
  ST_GeomFromText('POINT(116.292078 39.919622)'),
  ST_GeomFromText('POINT(116.286676 39.919593)');
```

Returned result:

```
+-----+
|_c0   |
+-----+
| 460.6965312526471 |
+-----+
```

1.6.12.8.6. ST_Area_Sphere

This topic describes the ST_Area_Sphere function and provides an example of using this function.

Function declaration:

```
double ST_Area_Sphere(st_geometry geo)
```

Description: This function uses the algorithm provided by AMAP to calculate the geodesic area of the input geometry. This function uses only ST_Polygon and ST_MultiPolygon as input parameters.

Example:

```
SELECT geospatial.ST_Area_Sphere(geospatial.ST_GeomFromText('POLYGON((116.259097 40.202114,116.259024 40.20199,116.258768 40.201662,116.258376 40.201341,116.258031 40.201036,116.257675 40.200734,116.257429 40.200656,116.257357 40.200562,116.257392 40.200051,116.257506 40.199433,116.257569 40.198586,116.257564 40.19756,116.257561 40.197372,116.257552 40.197036,116.257554 40.19675,116.257539 40.196647,116.257502 40.19653,116.257343 40.196389,116.257153 40.196276,116.256733 40.196071,116.25582 40.195646,116.255628 40.195611,116.255468 40.195653,116.255385 40.195742,116.255347 40.195849,116.255258 40.197143,116.255103 40.199576,116.255078 40.200585,116.251227 40.20059,116.251098 40.203978,116.259433 40.204111,116.259247 40.203079,116.259097 40.202114))))');
```

Returned result:

```
+-----+
|_c0   |
+-----+
| 353493.765625 |
+-----+
```

1.6.12.9. R-tree index functions

1.6.12.9.1. ST_BuildRTreeIndex

This topic describes the ST_BuildRTreeIndex function and provides an example of using this function.

Function declaration:

```
RTree ST_BuildRTreeIndex(string uniqueId, string geometryWkt)
```

Description: This function is a user-defined aggregate function (UDAF). It uses the unique ID and well-known text (WKT) string of each geometry as input parameters to create the R-tree index. This function must be used with other R-tree functions.

Example:

```
SELECT geospatial.ST_BuildRTreeIndex(id, shape) AS index FROM poi_sample;
```

1.6.12.9.2. ST_ContainsFromRTree

This topic describes the ST_ContainsFromRTree function.

Function declaration:

```
ST_ContainsFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that are contained by the geometry from the R-tree index. This function is used to accelerate the ST_Contains query.

1.6.12.9.3. ST_CrossesFromRTree

This topic describes the ST_CrossesFromRTree function.

Function declaration:

```
ST_CrossesFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that cross the geometry from the R-tree index. This function is used to accelerate the ST_Crosses query.

1.6.12.9.4. ST_EqualsFromRTree

This topic describes the ST_EqualsFromRTree function.

Function declaration:

```
ST_EqualsFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that equal the geometry from the R-tree index. This function is used to accelerate the ST_Equals query.

1.6.12.9.5. ST_IntersectsFromRTree

This topic describes the ST_IntersectsFromRTree function.

Function declaration:

```
ST_IntersectsFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that intersect with the geometry from the R-tree index. This function is used to accelerate the ST_Intersects query.

1.6.12.9.6. ST_OverlapsFromRTree

This topic describes the ST_OverlapsFromRTree function.

Function declaration:

```
ST_OverlapsFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that overlap with the geometry from the R-tree index. This function is used to accelerate the ST_Overlaps query.

1.6.12.9.7. ST_TouchesFromRTree

This topic describes the ST_TouchesFromRTree function.

Function declaration:

```
ST_TouchesFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that spatially touch the geometry from the R-tree index. This function is used to accelerate the ST_Touches query.

1.6.12.9.8. ST_WithinFromRTree

This topic describes the ST_WithinFromRTree function and provides an example of using this function.

Function declaration:

```
ST_WithinFromRTree(string uniqueId, string geometryWkt, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of objects that include the geometry from the R-tree index. This function is used to accelerate the ST_Within query.

Example:

Query the intersections of line segments in the A table and polygons in the B table.

```
set odps.sql.allow.cartesian=true;
SELECT a.id as link_id, b.id as shape_id
FROM link_sample_wkt a, poi_sample_wkt b
WHERE geospatial.ST_IsValid(b.shape)
AND geospatial.ST_Intersects(
    geospatial.ST_LineString(a.line),
    geospatial.ST_Multipolygon(b.shape));
```

Returned result:

```
Summary:
resource cost: cpu 3.28 Core * Min, memory 5.76 GB * Min
inputs:
meta_dev.poi_sample_wkt: 1000 (237592 bytes)
meta_dev.link_sample_wkt: 1000 (105940 bytes)
outputs:
Job run time: 111.000
+-----+-----+
| link_id | shape_id |
+-----+-----+
| 5121371185457659960 | B000A844XK |
| 5121377123249946651 | B000A85TV4 |
| 5121377166199619654 | B000A844KT |
+-----+-----+
```

After optimization by using the new function:

```
SELECT /*+mapjoin(i)*/
    geospatial.ST_IntersectsFromRTree(id, line, i.index)
    AS (link_id, shape_id)
FROM link_sample_wkt
JOIN
(
    SELECT geospatial.ST_BuildRTreeIndex(id, shape) AS index
    FROM poi_sample_wkt
    WHERE geospatial.ST_IsValid(shape)
) i;
```

Returned result:

```
Summary:
resource cost: cpu 1.03 Core * Min, memory 1.99 GB * Min
inputs:
meta_dev.poi_sample_wkt: 1000 (237592 bytes)
meta_dev.link_sample_wkt: 1000 (105940 bytes)
outputs:
Job run time: 41.000
+-----+-----+
| link_id | shape_id |
+-----+-----+
| 5121371185457659960 | B000A844XK |
| 5121377123249946651 | B000A85TV4 |
| 5121377166199619654 | B000A844KT |
+-----+-----+
```

1.6.12.9.9. ST_KNNFromRTree

This topic describes the ST_KNNFromRTree function and provides an example of using this function.

Function declaration:

```
ST_KNNFromRTree(string uniqueId, string geometryWkt, int k, RTree rtree)
```

Description: This function is a user-defined table-valued function (UDTF). It uses the unique ID and well-known text (WKT) string of each geometry and the R-tree index that is created by calling ST_BuildRTreeIndex as input parameters. This function returns the IDs of k objects that are near to the geometry from the R-tree index.

Example:

Create an R-tree for all points in a table and use the KNN function to find the nearest point of each point.

```
SELECT /*+mapjoin(i)*/
  geospatial.ST_KNNFromRTree(id, point, 1, i.index) AS (id1, id2)
FROM poi_sample_wkt
JOIN
(
  SELECT geospatial.ST_BuildRTreeIndex(id, point) AS index
  FROM poi_sample_wkt
) i;
```

Returned result:

```
Summary:
resource cost: cpu 1.17 Core * Min, memory 2.24 GB * Min
inputs:
  meta_dev.poi_sample_wkt: 1000 (237592 bytes)
outputs:
Job run time: 46.000
+----+----+
| id1 | id2 |
+----+----+
| B000A01B4E | B000A01B4E |
| B000A01C19 | B000A01C19 |
| B000A023A5 | B000A023A5 |
| B000A02F81 | B000A02F81 |
| B000A07BEE | B000A07BEE |
| B000A07E06 | B000A07E06 |
| B000A08863 | B000A08863 |
...
-- The table has 1,000 rows of data. This function returns 1,000 rows of data, which meets your expectations.
```

1.6.12.10. Other functions

1.6.12.10.1. ST_IsValid

This topic describes the ST_IsValid function and provides an example of using this function.

Function declaration:

```
boolean ST_IsValid(st_geometry geometry)
boolean ST_IsValid(string wkt)
```

Description: This function checks whether the input geometry or well-known text (WKT) string meets the requirements.

Example:

```
SELECT ST_IsValid('POINT(-102.849854 36.451113)');
SELECT ST_IsValid(ST_Point('POINT(-102.849854 36.451113)'));
```

1.6.12.10.2. ST_Transform

This topic describes the ST_Transform function and provides an example of using this function.

Function declaration:

```
st_geometry ST_TransformWGS84(st_geometry geometry)
st_geometry ST_Transform(st_geometry geometry, integer toSRID)
st_geometry ST_Transform(st_geometry geometry, integer fromSRID, integer toSRID)
```

Description: This function converts the coordinates of the input geometry from one spatial reference system to another. The ST_TransformWGS84 function converts the coordinates of the geometry from EPSG:4326 to EPSG:3857. The ST_Transform function converts the geometry from fromSRID to toSRID. If the overload function contains only toSRID, you must call the ST_SetSRID function first.

Example:

```
SELECT ST_AsText(ST_Transform(ST_GeomFromText('POLYGON((743238 2967416,743238 2967450, 743265 2967450,743265.625 2967416,743238 2967416))', 2249, 4326)));
SELECT ST_AsText(ST_TransformWGS84(ST_GeomFromText('POLYGON((-71.1776848522251 42.3902896512902,-71.1776843766326 42.3903829478009, -71.1775844305465 42.3903826677917,-71.1775825927231 42.3902893647987,-71.1776848522251 42.3902896512902))')));
```

1.6.13. MaxCompute Hash Clustering

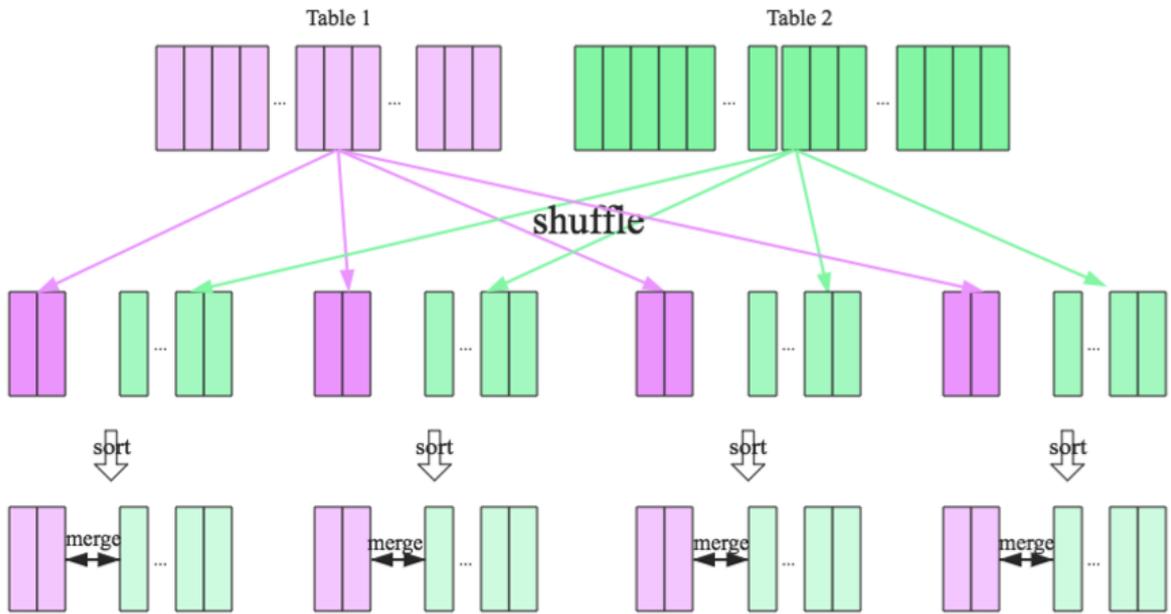
1.6.13.1. Background information

This topic describes the background of MaxCompute Hash Clustering.

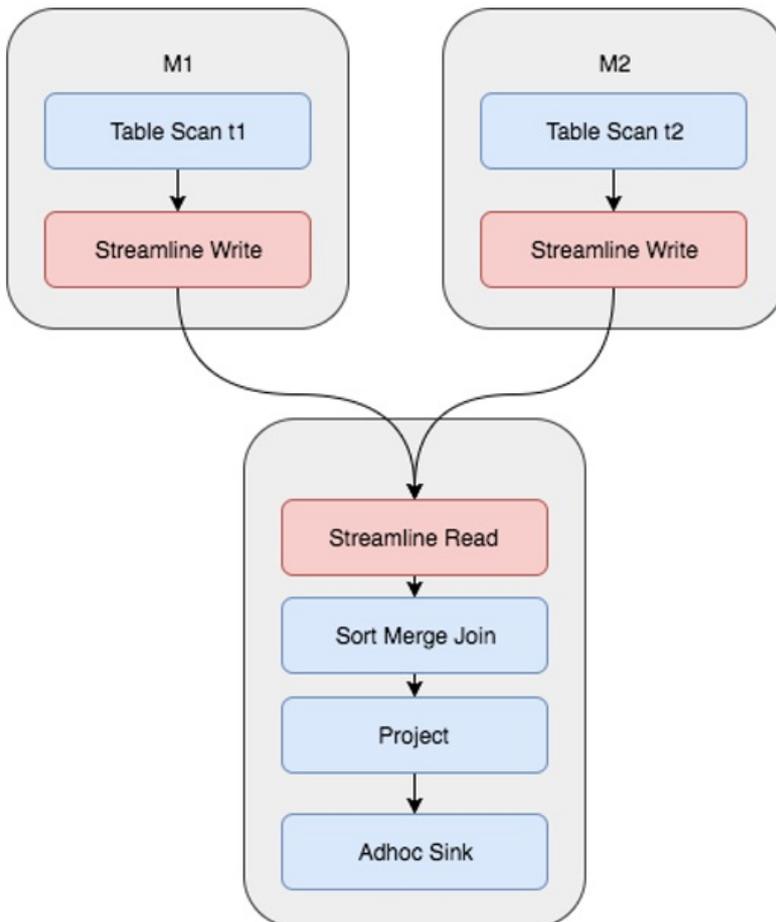
JOIN operations are commonly used for queries in MaxCompute. MaxCompute provides the following methods to implement JOIN operations:

- Broadcast hash join: This method is used when a JOIN operation involves a small table. The small table is broadcasted and transferred to all instances in a join task. Then, the hash join operation is performed to join the small table with a large table.
- Shuffle hash join: This method is used when a JOIN operation involves large tables that cannot be directly broadcasted. In this case, the hash shuffle operation is performed on two tables based on join keys. The hash results for the same key-value pairs are the same. This ensures that results that have the same key are collected to the same instance of a join task. For each instance, a hash table is created by using a small table, probe operations are performed by using a large table, and then the tables are joined.
- Sort merge join: This method is used when a JOIN operation involves larger tables and the preceding methods cannot be used because the memory is insufficient to create a hash table. In this case, the hash shuffle operation is performed on two tables based on join keys, the obtained values are sorted by using

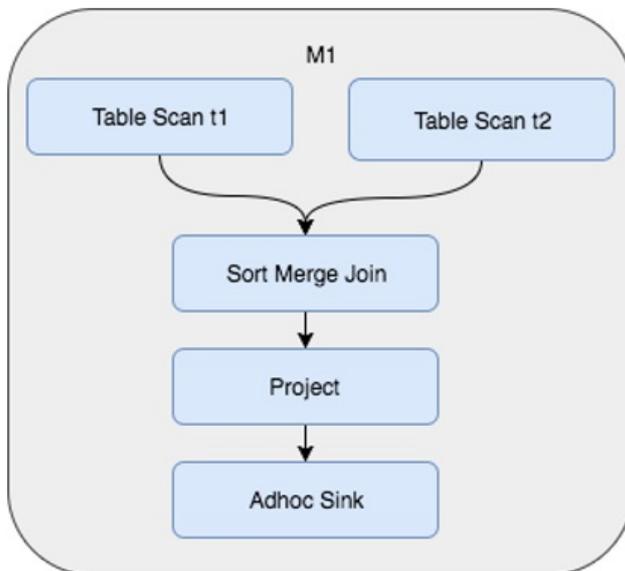
join keys, and then the sorted values are merged.



The sort merge join method is commonly used in MaxCompute because MaxCompute processes large amounts of data in most cases. If you use this method, shuffle and join operations are repeatedly performed. The physical execution plan of Job Scheduler of a JOIN operation requires multiple stages, which consumes excessive amounts of resources.



To resolve this issue, MaxCompute allows you to configure the hash shuffle and sort attributes when the data is initially generated in a table. This prevents data from being repeatedly shuffled and sorted in subsequent queries. The number of stages in the physical execution plan of Job Scheduler of a JOIN operation is also reduced. The preceding figure shows that only one stage is required.



MaxCompute Hash Clustering allows you to configure the shuffle and sort attributes of a table when you create the table. Then, MaxCompute optimizes the execution plan, improves the efficiency, and saves resources based on the existing storage characteristics.

1.6.13.2. Descriptions

1.6.13.2.1. Enable Hash Clustering

This topic describes how to enable the Hash Clustering feature of MaxCompute.

The Hash Clustering feature has been launched. By default, this feature is enabled. If you want to use clustered indexes, add the following flag:

```
set odps.sql.cfile2.enable.read.write.index.flag=true;
```

After the flag is set to true, the system automatically creates indexes for the sorted hash buckets to improve query efficiency. To use clustered indexes, you must add this flag during table creation and subsequent queries. If you want to use clustered indexes in your project at all times, contact the MaxCompute team.

Note Clustered indexes improve the efficiency of queries (equivalent values or ranges) based on sort keys. You can experience the superior performance provided by Hash Clustering even if you do not add this flag.

1.6.13.2.2. Create a hash-clustered table

This topic describes how to create a hash-clustered table.

You can use the following statement to create a hash-clustered table. You must specify cluster keys or hash keys and the number of hash buckets. The sort operation is optional. However, we recommend that you use the same parameter values as the cluster keys to achieve optimal performance.

```
CREATE TABLE [IF NOT EXISTS] table_name
  [(col_name data_type [comment col_comment], ...)]
  [comment table_comment]
  [PARTITIONED BY (col_name data_type [comment col_comment], ...)]
  [CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...])] INTO number_of_buckets BUCKETS]
  [AS select_statement]
```

You can use the following statement to create a standard table:

```
CREATE TABLE T1 (a string, b string, c bigint) CLUSTERED BY (c) SORTED BY (c) INTO 1024 BUCKETS;
```

You can use the following statement to create a partitioned table:

```
CREATE TABLE T1 (a string, b string, c bigint) PARTITIONED BY (dt string) CLUSTERED BY (c) SORTED BY (c) INTO 1024 BUCKETS;
```

The following sections detail the CLUSTERED BY, SORTED BY, and INTO number_of_buckets BUCKETS clauses.

CLUSTERED BY

The CLUSTERED BY clause specifies hash keys. MaxCompute performs the hash operation on the specified column and distributes data to buckets based on the hash values. To prevent data skew and hot spots, and to concurrently execute statements, we recommend that you specify a column that has large value ranges and a small number of duplicate key-value pairs in CLUSTERED BY. In addition, to optimize the JOIN operation, we recommend that you select commonly used join or aggregation keys. The join and aggregation keys are similar to the primary keys in conventional databases.

SORTED BY

The SORTED BY clause specifies how fields are sorted in a bucket. We recommend that you specify the same column in SORTED BY as that in CLUSTERED BY to improve execution efficiency. After you specify the column in SORTED BY, MaxCompute automatically generates indexes and then executes SQL statements faster when you query data based on these indexes.

INTO number_of_buckets BUCKETS

The INTO number_of_buckets BUCKETS clause specifies the number of hash buckets, which is required. The number of hash buckets is determined by the volume of data. More buckets indicate higher concurrency, which shortens the job running time. However, if a large number of buckets exist, excessive small files may be generated. In addition, high concurrency increases CPU time. We recommend that you set the volume of data for each bucket to a value that ranges from 500 MB to 1 GB. If a large table is used, you can adjust the value to a larger value as required.

You can remove the shuffle operation only for tables with the same number of buckets in MaxCompute. In later versions, MaxCompute will support bucket alignment. You will be able to remove the shuffle operation for tables whose numbers of buckets are multiples or factors of each other. To achieve bucket alignment, we recommend that you set the number of buckets to a power of 2, for example, 512, 1,024, and 2,048. The maximum number of buckets is 4,096. If the number of buckets exceeds the value, the performance and resource usage may be affected.

If you want to remove the shuffle and sort operations during a JOIN operation on two tables, the numbers of hash buckets in the tables must be the same. If the numbers that are calculated based on the aforementioned method are inconsistent, we recommend that you use the larger number for the JOIN operation. This ensures that SQL statements can be concurrently executed in an efficient manner.

If the sizes of two tables greatly differ, you can set the number of buckets for the large table to several times of that for the small table, for example, 256 and 1,024. If automatic hash bucket split and merging are supported, the settings can be optimized by using data features.

1.6.13.2.3. Modify table attributes

This topic describes how to modify the attributes of a hash-clustered table.

For a partitioned table, MaxCompute allows you to execute the ALTER TABLE statement to add the Hash Clustering attribute to a table or remove the Hash Clustering attribute from a table.

```
ALTER TABLE table_name
  [CLUSTERED BY (col_name [, col_name, ...]) [SORTED BY (col_name [ASC | DESC] [, col_name [ASC | DESC] ...])] INTO
  number_of_buckets BUCKETS]
ALTER TABLE table_name NOT CLUSTERED;
```

When you use the ALTER TABLE statement, take note of the following points:

- The ALTER TABLE statement can modify only the Hash Clustering attribute of a partitioned table. The Hash Clustering attribute cannot be modified after it is added to a non-partitioned table.
- The ALTER TABLE statement takes effect only on the new partitions of a table, which include the partitions generated by using the INSERT OVERWRITE statement. New partitions are stored based on the Hash Clustering attribute. The storage formats of existing partitions remain unchanged.
- The ALTER TABLE statement takes effect only on the new partitions of a table. Therefore, you cannot specify a partition in this statement.

The ALTER TABLE statement is suitable for existing tables. After the Hash Clustering attribute is added, new partitions are stored based on the Hash Clustering attribute.

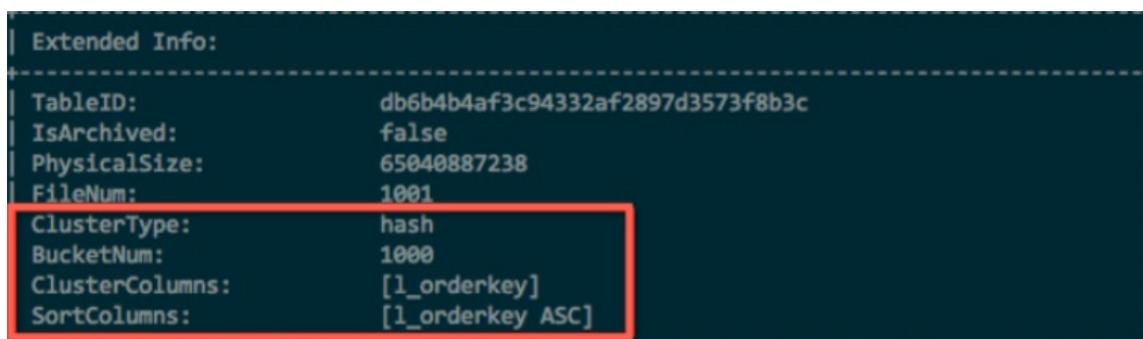
1.6.13.2.4. View and verify table attributes

This topic describes how to view and verify attributes of a hash-clustered table.

After you create a hash-clustered table, execute the following statement to view table attributes:

```
DESC EXTENDED table_name;
```

The table attributes are displayed in Extended Info.

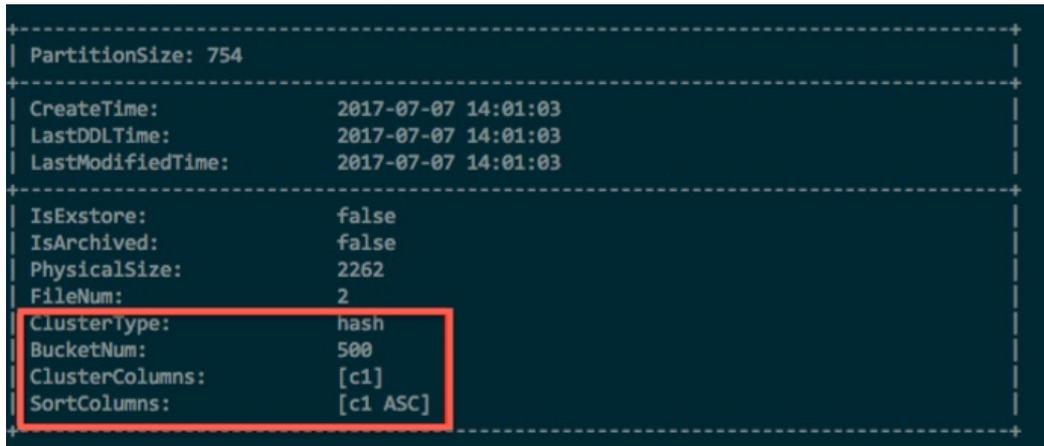


```
Extended Info:
-----
TableID:          db6b4b4af3c94332af2897d3573f8b3c
IsArchived:       false
PhysicalSize:     65040887238
FileNum:          1001
ClusterType:      hash
BucketNum:        1000
ClusterColumns:   [1_orderkey]
SortColumns:      [1_orderkey ASC]
```

You can also execute the following statement to view partition attributes of a partitioned table:

```
DESC EXTENDED table_name partition(pt_spec);
```

The following figure shows the execution result.



1.6.13.3. Benefits

1.6.13.3.1. Bucket pruning and index optimization

This topic describes bucket pruning and index optimization.

The following code provides a syntax sample:

```
CREATE TABLE t1 (id bigint, a string, b string) CLUSTERED BY (id) SORTED BY (id) INTO 1000 BUCKETS;
...
SELECT t1.a, t1.b, t1.c FROM t1 WHERE t1.id=12345;
```

This syntax indicates a full scan for a standard table. A full scan for a large table consumes a large number of resources. However, if the hash shuffle operation is performed on all id fields and the id fields are sorted, the query is greatly simplified.

Sample procedure:

1. Find the hash bucket that corresponds to 12345. This query is performed in only one bucket, not all 1,000 buckets. This process is called bucket pruning.
2. Data in a bucket is stored based on IDs. MaxCompute automatically creates indexes and uses the INDEX LOOKUP function to locate relevant records.

The simplified procedure not only greatly reduces the number of mappers, but also allows mappers to locate the page where the data is stored by using the INDEX function. Therefore, the volume of loaded data is greatly reduced.

1.6.13.3.2. Aggregation optimization

This topic describes aggregation optimization.

The following code provides an example:

```
SELECT department, SUM(salary) FROM employee GROUP BY (department);
```

In most cases, the department column is shuffled and sorted. Then, a stream aggregate operation is performed to collect statistics on the department groups. However, if `CLUSTERED BY (department) SORTED BY (department)` is executed for the table data, the shuffle and sort operations are no longer required.

1.6.13.3.3. Storage optimization

This topic describes storage optimization.

In addition to computation optimization, storage space is greatly saved if tables are shuffled and stored in a sorted manner. MaxCompute uses the column store at the underlying layer. Records that have the same or similar key-value pairs are stored together by the sort function, which facilitates encoding and compression. This way, compression efficiency is improved. In some extreme cases, a sorted table can save 50% more storage space than an unsorted table. Therefore, Hash Clustering is suitable for the storage of tables that have long lifecycles.

The following figure shows a sample table with 100 GB of TPC-H line items and multiple data types, such as INT, DOUBLE, and STRING. When Hash Clustering is used, about 10% of the storage space is saved while the volume of data and compression format remain unchanged.

```

CreateTime:          2016-04-17 21:48:08
LastDDLTime:        2016-04-17 21:48:08
LastModifiedTime:   2016-04-17 21:50:10
-----
InternalTable: YES  | Size: 23573055432
-----
Native Columns:
-----
Field      | Type      | Label | Comment
-----
l_orderkey | bigint    |       |
l_partkey  | bigint    |       |
l_suppkey  | bigint    |       |
l_linenumb | bigint    |       |
l_quantity | double    |       |
l_extended | double    |       |
l_discount | double    |       |
l_tax      | double    |       |
l_returnfl | string    |       |
l_linestat | string    |       |
l_shipdat  | string    |       |
l_commitda | string    |       |
l_receiptd | string    |       |
l_shipinst | string    |       |
l_shipmod  | string    |       |
l_comment  | string    |       |
    
```

```
| CreateTime:          2017-07-13 14:40:11
| LastDDLTime:        2017-07-13 14:40:11
| LastModifiedTime:   2017-07-13 15:05:04
+-----+-----+
| InternalTable: YES  | Size: 21658913950
+-----+-----+
| Native Columns:
+-----+-----+
| Field      | Type   | Label | Comment
+-----+-----+
| l_orderkey | bigint |       |
| l_partkey  | bigint |       |
| l_suppkey  | bigint |       |
| l_linenumb | bigint |       |
| l_quantity | double |       |
| l_extended | double |       |
| l_discount | double |       |
| l_tax      | double |       |
| l_returnfl | string |       |
| l_linestat | string |       |
| l_shipdat | string |       |
| l_commitda | string |       |
| l_receiptd | string |       |
| l_shipinsh | string |       |
| l_shipmod | string |       |
| l_comment  | string |       |
```

1.6.13.4. ShuffleRemove

This topic describes the operations that are related to ShuffleRemove.

- Range-clustered tables support the join and aggregate operations. If a join key or group key is a range-clustered key or its prefix, data redistribution is not required. This mechanism is called ShuffleRemove, which improves execution efficiency.

Usage: The `odps.optimizer.enable.range.partial.repartitioning` flag specifies whether to enable this feature. By default, this feature is disabled.

- If the two hash-clustered tables you want to join have different numbers of buckets but the numbers are multiples of each other, data redistribution is not required. This improves execution efficiency.

Usage: The `odps.optimizer.enable.hash.partial.repartitioning` flag specifies whether to enable this feature. By default, this feature is enabled.

- Correlated Shuffle Remove is supported. If data meets distribution requirements but does not meet the sorting requirements, you can add a sort operator to avoid data redistribution.

1.6.13.5. Limits

This topic describes limits on MaxCompute Hash Clustering.

Limits:

- The INSERT INTO statement is not supported. You can only execute the INSERT OVERWRITE statement to add data.
- Small files cannot be merged. Most data is evenly distributed in buckets when it is split. Therefore, the number of small files is not large. If you merge files, the data distribution is affected. However, you can still

use the merge and archive commands to change the storage format of a table file and the format of a RAID file.

- You cannot use Tunnel to upload data to a range-clustered table because the data uploaded by using Tunnel is unsorted.

1.6.14. Common MaxCompute SQL parameter settings

1.6.14.1. Map configurations

This topic describes Map configurations.

```
set odps.sql.mapper.cpu=100;
```

Description: specifies the number of CPUs used by each instance in a Map task. Default value: 100. Valid values: [50,800].

```
set odps.sql.mapper.memory=1024;
```

Description: specifies the memory size of each instance in a Map task. Unit: MB. Default value: 1024. Valid values: [256,12288].

```
set odps.sql.mapper.merge.limit.size=64;
```

Description: specifies the maximum size of control files to be merged. Unit: MB. Default value: 64. Valid values: [0,Integer.MAX_VALUE]. You can configure this variable to control the input of mappers.

```
set odps.sql.mapper.split.size=256;
```

Description: specifies the maximum data input volume for a Map task. Unit: MB. Default value: 256. Valid values: [1,Integer.MAX_VALUE]. You can configure this variable to control the input of mappers.

1.6.14.2. Join configurations

This topic describes Join configurations.

```
set odps.sql.joiner.instances=-1;
```

Description: specifies the number of instances of a join task. Default value: -1. Valid values: [0,2000].

```
set odps.sql.joiner.cpu=100;
```

Description: specifies the number of CPUs used by each instance of a join task. Default value: 100. Valid values: [50,800].

```
set odps.sql.joiner.memory=1024;
```

Description: specifies the memory size of each instance of a join task. Unit: MB. Default value: 1024. Valid values: [256,12288].

1.6.14.3. Reduce configurations

This topic describes Reduce configurations.

```
set odps.sql.reducer.instances=-1;
```

Description: specifies the number of instances in a Reduce task. Default value: -1. Valid values: [0,2000].

```
set odps.sql.reducer.cpu=100;
```

Description: specifies the number of CPUs used by each instance in a Reduce task. Default value: 100. Valid values: [50,800].

```
set odps.sql.reducer.memory=1024;
```

Description: specifies the memory size of each instance in a Reduce task. Unit: MB. Default value: 1024. Valid values: [256,12288].

1.6.14.4. UDF configurations

This topic describes the configurations of user-defined functions (UDFs).

```
set odps.sql.udf.jvm.memory=1024;
```

Description: specifies the maximum memory size used by the UDF JVM heap. Unit: MB. Default value: 1024. Valid values: [256,12288].

```
set odps.sql.udf.timeout=600;
```

Description: specifies the timeout period of a UDF. Unit: seconds. Default value: 600. Valid values: [0,3600].

```
set odps.sql.udf.python.memory=256;
```

Description: specifies the maximum memory size used by the UDF Python API. Unit: MB. Default value: 256. Valid values: [64,3072].

```
set odps.sql.udf.optimize.reuse=true/false;
```

Description: When this parameter is set to true, each UDF function expression can be calculated only once. This improves performance. Default value: true.

```
set odps.sql.udf.strict.mode=false/true;
```

Description: specifies whether functions return NULL or an error if dirty data is found. If the parameter is set to true, an error is returned. Otherwise, NULL is returned.

1.6.14.5. MapJoin configurations

This topic describes MapJoin configurations.

```
set odps.sql.mapjoin.memory.max=512;
```

Description: specifies the maximum memory size for a small table that is involved in a MapJoin operation. Unit: MB. Default value: 512. Valid values: [128,2048].

```
set odps.sql.reshuffle.dynamicpt=true/false;
```

Description:

- Dynamic partitioning is time-consuming in some scenarios. Disabling dynamic partitioning can accelerate SQL.
- If the number of dynamic partitions is small, disabling dynamic partitioning can prevent data skew.

1.6.14.6. Configurations of data skew

This topic describes configurations of data skew.

```
set odps.sql.groupby.skewindata=true/false;
```

Description: allows you to enable GROUP BY optimization.

```
set odps.sql.skewjoin=true/false;;
```

Description: allows you to enable JOIN optimization. This is effective only when odps.sql.skewinfo is specified.

```
set odps.sql.skewinfo;
```

Description: allows you to configure detailed information for JOIN optimization. Command syntax:

```
set odps.sql.skewinfo=skewed_src:(skewed_key)[("skewed_value")]
```

Example:

Configure a single skewed value for a single field.

```
set odps.sql.skewinfo=src_skewjoin1:(key)[("0")];
-- Output result: explain select a.key c1, a.value c2, b.key c3, b.value c4 from src a join src_skewjoin1 b on a.key = b.key;
```

Configure multiple skewed values for a single field.

```
set odps.sql.skewinfo=src_skewjoin1:(key)[("0")("1")];
-- Output result: explain select a.key c1, a.value c2, b.key c3, b.value c4 from src a join src_skewjoin1 b on a.key = b.key;
```

1.6.15. MapReduce-to-SQL conversion for execution

1.6.15.1. Overview

This topic provides an overview of the MapReduce-to-SQL conversion for execution feature.

MaxCompute provides a series of Java APIs for MapReduce to process data.

MapReduce programs are automatically converted to SQL for execution. After the conversion, you can use the compiler, cost-based optimizer, and vectorized execution engine released with MaxCompute V2.0 to process the MapReduce programs. The new features of the SQL engine can also be used. The features, performance, and stability of the SQL engine are optimized.

 **Note**

- You do not need to change the original APIs and job logic.
- Only MapReduce jobs of the OpenMR type, which are written by using MapReduce APIs, can be converted to SQL.
- This feature can be used only for projects and jobs.
- This feature supports views as the input.
- This feature supports external tables as the input.
- This feature supports TemporaryFile reads and writes.
- This feature allows you to read data from and write data to hash-clustered tables.
- This feature supports near-real-time execution of small jobs.

1.6.15.2. Configure the MaxCompute client

This topic describes how to configure the MaxCompute client.

1. Download the latest **MaxCompute client** package to your computer and configure the client.

 **Note** On the download page, you can obtain the MaxCompute client package of the required version in the **Other Resources** section.

2. Configure the execution mode.

You can configure the execution mode based on your business requirements. The default execution mode is lot. In lot mode, jobs are run by MapReduce. The new compiler, optimizer, and execution engine are not used.

You can enable the execution mode by setting the `odps.mr.run.mode` parameter. Valid values: lot, sql, and hybrid.

- Method 1: Enable the execution mode at the project level. In this case, the configuration takes effect on all jobs in the project. Therefore, only the project administrator can apply to enable the execution mode. Set the `odps.mr.run.mode` parameter to hybrid or sql. If SQL execution fails in hybrid mode, the job is run by MapReduce. If SQL execution fails in sql mode, an error is reported.
- Method 2: Enable the execution mode at the session level. This method applies only to the current job. To enable the execution mode, use one of the following methods:
 - Add a set flag, such as `set odps.mr.run.mode=hybrid;` before JAR statements.
 - Configure job parameters. Example:

```
JobConf job = new JobConf();  
job.set("odps.mr.run.mode","hybrid")
```

MaxCompute O&M personnel can enable the execution mode at the project level at a later point in time.

1.6.15.3. Configure running settings in DataWorks

This topic describes how to configure running settings in DataWorks.

Jobs that run in DataWorks are updated by the O&M personnel of MaxCompute and DataWorks. You do not need to update your client.

- Enable the conversion for a single job.

You can add a set flag before the statement of a MapReduce job or configure job parameters for the set flag. These methods take effect at the session level and apply only to the current job.

The following examples show how to use these methods:

- Add a set flag, such as `set odps.mr.run.mode=hybrid;` before JAR statements.
- Configure job parameters. Example:

```
JobConf job = new JobConf();
job.set("odps.mr.run.mode", "hybrid")
```

- Enable the conversion at the project level by setting `odps.mr.run.mode` for a project.

1.6.15.4. View details

This topic describes how to view MapReduce-to-SQL conversion results and running details of SQL jobs.

You can use Logview and MaxCompute Studio to view MapReduce-to-SQL conversion results and running details of SQL jobs.

- Logview XML

Open Logview and click the LOT node in the center of the page. The SQL jobs that are converted from MapReduce jobs are included in the XML information about the node. Example:

```
create temporary function mr2sql_mapper_152955927079392291755 as 'com.aliyun.odps.mapred.bridge.LotMapperUDTF' using ;
create temporary function mr2sql_reducer_152955927079392291755 as 'com.aliyun.odps.mapred.bridge.LotReducerUDTF' using ;
@sub_query_mapper :=
SELECT k_id,v_gmt_create,v_gmt_modified,v_product_id,v_admin_seq,v_sku_attr,v_sku_price,v_sku_stock,v_sku_code,v_sku_image,v_delivery_time,v_sku_bulk_order,v_sku_bulk_discount,v_sku_image_version,v_currency_code
FROM(
SELECT mr2sql_mapper_152955927079392291755(id,gmt_create,gmt_modified,product_id,admin_seq,sku_attr,sku_price,sku_stock,sku_code,sku_image,delivery_time,sku_bulk_order,sku_bulk_discount,sku_image_version,currency_code) as (k_id,v_gmt_create,v_gmt_modified,v_product_id,v_admin_seq,v_sku_attr,v_sku_price,v_sku_stock,v_sku_code,v_sku_image,v_delivery_time,v_sku_bulk_order,v_sku_bulk_discount,v_sku_image_version,v_currency_code)
FROM ae_antispam.product_sku_tt_inc
WHERE ds = "20180615" AND hh = "21"
UNION ALL
SELECT mr2sql_mapper_152955927079392291755(id,gmt_create,gmt_modified,product_id,admin_seq,sku_attr,sku_price,sku_stock,sku_code,sku_image,delivery_time,sku_bulk_order,sku_bulk_discount,sku_image_version,currency_code) as (k_id,v_gmt_create,v_gmt_modified,v_product_id,v_admin_seq,v_sku_attr,v_sku_price,v_sku_stock,v_sku_code,v_sku_image,v_delivery_time,v_sku_bulk_order,v_sku_bulk_discount,v_sku_image_version,v_currency_code)
FROM ae_antispam.product_sku
) open_mr_alias1
DISTRIBUTE BY k_id SORT BY k_id ASC;
@sub_query_reducer :=
SELECT mr2sql_reducer_152955927079392291755(k_id,v_gmt_create,v_gmt_modified,v_product_id,v_admin_seq,v_sku_attr,v_sku_price,v_sku_stock,v_sku_code,v_sku_image,v_delivery_time,v_sku_bulk_order,v_sku_bulk_discount,v_sku_image_version,v_currency_code) as (id,gmt_create,gmt_modified,product_id,admin_seq,sku_attr,sku_price,sku_stock,sku_code,sku_image,delivery_time,sku_bulk_order,sku_bulk_discount,sku_image_version,currency_code)
FROM @sub_query_mapper;
FROM @sub_query_reducer
INSERT OVERWRITE TABLE ae_antispam.product_sku
SELECT id,gmt_create,gmt_modified,product_id,admin_seq,sku_attr,sku_price,sku_stock,sku_code,sku_image,delivery_time,sku_bulk_order,sku_bulk_discount,sku_image_version,currency_code;
```

- Logview details or summary

A new execution engine is used to run jobs.

```
Job run mode: fuxi job
Job run engine: execution engine
```

- Logview details or JSON summary

The JSON summary information in MapReduce contains only the input and output information about Map and Reduce. However, the JSON summary information in SQL allows you to view details about each stage of SQL execution, such as all execution parameters, logical execution plans, physical execution plans, and execution details. Example:

```
"midlots" :
[
  "LogicalTableSink(table=[[odps_fighting.flt_20180621104445_step1_ad_quality_tech_qp_algo_antifake_word
bag_filter_bag_change_result_lv2_20, auctionid,word,match_word(3) {0, 1, 2}]]
OddsLogicalProject(auctionid=[${0}], word=[${1}], match_word=[${2}]
OddsLogicalProject(auctionid=[${0}], word=[${1}], match_word=[${2}]
OddsLogicalProject(auctionid=[${0}], word=[${1}], match_word=[${2}]
OddsLogicalProject(auctionid=[${2}], word=[${3}], match_word=[${4}]
OddsLogicalTableFunctionScan(invocation=[[MR2SQL_MAPPER_152955294118813063732(${0}, ${1})]()], rowType=[
RecordType(VARCHAR(2147483647) item_id, VARCHAR(2147483647) text, VARCHAR(2147483647) __tf_0_0, VARCH
AR(2147483647) __tf_0_1, VARCHAR(2147483647) __tf_0_2)])
OddsLogicalTableScan(table=[[ad_quality_tech_qp_algo_antifake_wordbag_filter_bag_change_lv2_20, item_id,
text(2) {0, 1}]]
]
```

1.6.15.5. Perform operations on the distributed file system

This topic describes how to perform operations on the distributed file system.

Procedure

1. Specify volume files.

You can use one of the following methods to specify volume files:

- o Use a utility class to specify the input and output files:

```
com.aliyun.odps.mapred.utils.InputUtils.addVolume( new VolumeInfo([project,]inVolume,inPartition, "inLa
bel"), new JobConf());
com.aliyun.odps.mapred.utils.OutputUtils.addVolume( new VolumeInfo([project,]outVolume, outPartition,
"outLabel"), new JobConf());
```

In the preceding commands, project and label are optional, and the current project and default label are used by default. If multiple input and output files are used, labels are used to distinguish the files from each other. Authorization is required before you access the volume files of other projects.

- o Configure parameters to specify the volume and partition of the input and output files. If multiple input or output files are used, separate the parameters with commas (,).

```
set odps.sql.volume.input[/output].desc = [<project>.<table>.<partition>[:<label>];
```

2. Call the following method by using a context object in the map and reduce steps to write data to the distributed file system or write data stream input and output files:

```
context.getOutputVolumeFileSystem();
```

1.6.16. Analysis of the mapping between SQL input and output fields

1.6.16.1. Overview

This topic describes the feature of analyzing the mapping between fields in input and output tables.

Example:

```
select key, sum(value) as total from src group by key;
```

The result that is shown in the following figure is returned.

```
= Column Lineage
Column : key
Source Columns :
  test2.src.key

Column : total
Source Columns :
  test2.src.value
Functions :
  sum(test2.src.value)
```

Two columns are returned: key and total. The key column corresponds to the src.key column of the input table. The total column corresponds to the src.value column of the input table.

1.6.16.2. Usage notes

This topic provides an example on how to use the feature of analyzing the mapping between fields in input and output tables.

Output format

Field mapping analysis supports human-readable and JSON formats. You can use the `set odps.sql.select.output.format=HumanReadable/json` flag to specify the output format.

SDK-based field mapping analysis

Example

```
Odps odps = initOdps();
// To perform analysis, use LineageTask.
LineageTask task = new LineageTask("task_name", "select * from dual;");
// Optional. Use the preceding flag to specify the output format.
Map<String, String> settings = new LinkedHashMap<>();
settings.putIfAbsent("odps.sql.select.output.format", "json");
task.setProperty("settings", JSON.toJSONString(settings));
// Submit code to the server for field mapping analysis.
Instance instance = odps.instances().create(task);
System.out.println(instance.getId());
String logView = odps.logview().generateLogView(instance, 72);
System.out.println(logView);
instance.waitForSuccess();
// Obtain the analysis result.
System.out.println(instance.getTaskResults().get("task_name"));
```

odpscmd-based field mapping analysis

Examples

CLI mode: Use the `-X` parameter for field mapping analysis.

```
./bin/odpscmd.bat -X D:\lineage.q
```

Interactive mode: After you enter the interactive mode of odpscmd, you can use the preceding flag to specify the output format. The usage method is similar to that used to submit SQL jobs.

```
lineage_test>set odps.sql.task.mode=LINEAGE;
-- OK is returned. Continue to run the following command:
odps@ lineage_test>set odps.sql.select.output.format=Humanreadable;
-- OK is returned. Continue to run the following command:
lineage_test>select * from dual;
```

Returned result:

```
== Column Lineage
Column : id
Source Columns :
test2.dual.id
```

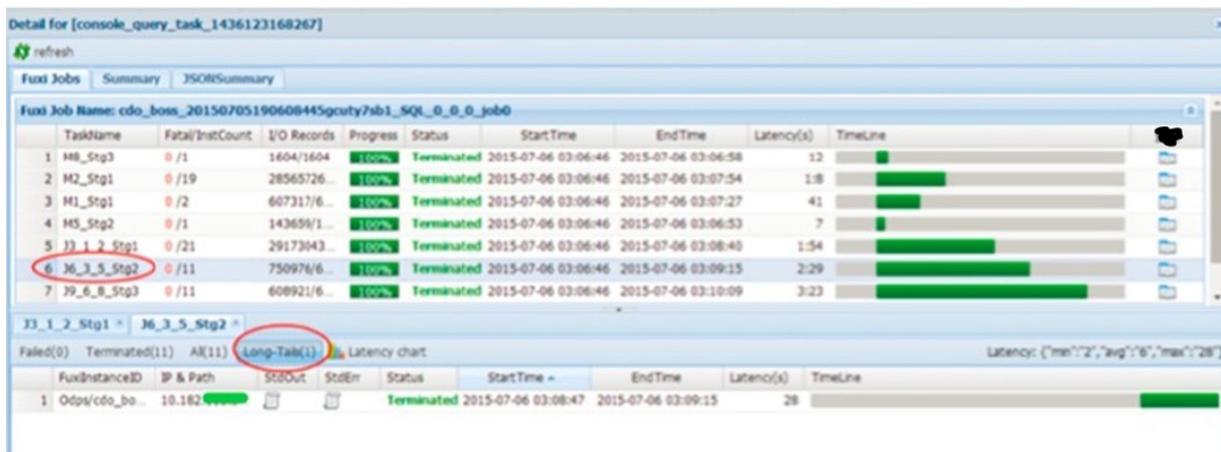
1.6.17. Common MaxCompute SQL errors and solutions

1.6.17.1. Data skew

1.6.17.1.1. Overview

This topic provides an overview of data skew issues.

In most cases, data skew occurs if the min, max, and avg values of the instance time, input records, and output records parameters are unbalanced in an instance on which jobs run. For example, if the max value is significantly larger than the avg value, data skew may occur. You can use Logview to identify data skew issues, as shown in the following figure.



The Long Tails tab of each task displays the instances on which data skew occurs. The root cause of data skew is that the amount of data processed by a specific instance far exceeds that processed by other instances. As a result, the running time of this instance is also much longer than the average running time of other instances, and the entire job slows down.

Different types of data skew issues in SQL are resolved by using different methods. The subsequent topics describe possible causes of data skew issues and provide solutions.

1.6.17.1.2. GROUP BY data skew

This topic describes possible causes of the GROUP BY data skew and provides a solution.

Possible cause: GROUP BY keys are unevenly distributed, which results in data skew on reducers.

Solution: Set the anti-skew parameter before you execute SQL statements.

```
set odps.sql.groupby.skewindata=true;
```

 **Note** If you set this parameter to true, when the Shuffle hash algorithm is used, the system automatically adds a random factor and introduces a new task to prevent data skew.

1.6.17.1.3. DISTRIBUTE BY data skew

This topic describes possible causes of the DISTRIBUTE BY data skew and provides a solution.

Possible cause: Constants are used to execute the DISTRIBUTE BY clause for full sorting of the entire table. This results in data skew on reducers.

Solution: Avoid the preceding operation.

1.6.17.1.4. Data skew caused by JOIN operations

This topic describes the data skew caused by JOIN operations and provides a solution.

Possible cause: JOIN ON keys are unevenly distributed. For example, a key has a large number of duplicates in multiple tables that you want to join. This results in a Cartesian explosion of data in the instance on which the JOIN statement is executed and causes data skew.

Solution: You can use one of the following methods to resolve the data skew issue based on scenarios:

- If one of the tables that you want to join is a small table, execute the MAP JOIN statement, instead of the JOIN statement.
- Use separate logic to handle skewed keys. For example, if data skew occurs because a large number of null key values exist in both tables, you must filter out these null key values or use a CASE WHEN expression to replace the null key values with random values before you perform a JOIN operation.
- If you do not want to change the SQL statement, configure the following parameters to enable automatic optimization on MaxCompute:

```
set odps.sql.skewinfo=tab1:(col1,col2)[(v1,v2),(v3,v4),...];  
set odps.sql.skewjoin=true;
```

1.6.17.1.5. Data skew caused by multiple DISTINCT operations

This topic describes the data skew caused by multiple DISTINCT operations and provides a solution.

Possible cause: Multiple DISTINCT operations aggravate the GROUP BY skew.

Solution: Execute two-layer GROUP BY statements to alleviate the data skew issue. In most cases, do not use multiple DISTINCT operations at the same time.

1.6.17.1.6. Data skew caused by misuse of dynamic partitions

This topic describes the data skew caused by misuse of dynamic partitions and provides a solution.

Possible cause: In scenarios where dynamic partitions are used, if K Map instances and N destination partitions are deployed, $K \times N$ small files may be generated. This significantly increases the management workload of the file system. Therefore, the following configuration takes effect by default:

```
set odps.sql.reshuffle.dynamicpt=true;
```

An additional level of Reduce task is introduced. The same or a small number of other Reduce instances are used to write data to the same destination partition. This prevents a large number of small files from being generated. However, dynamic partition shuffle may cause data skew.

Solution: If the number of destination partitions is small, only an appropriate number of small files are generated. In this case, you can run the following command to set `odps.sql.reshuffle.dynamicpt` to false to prevent the data skew caused by dynamic partition shuffle.

```
set odps.sql.reshuffle.dynamicpt=false;
```

If the number of destination partitions is large, we recommend that you do not use dynamic partitions because a large number of small files are generated.

1.6.17.1.7. Use SKEWJOIN HINT to avoid skewed hot key values

This topic describes how to avoid skewed hot key values by using SKEWJOIN HINT. It also provides examples for reference.

Methods

- Method 1: Specify the name of a table in SKEWJOIN HINT. Note that the name specified in the hint is the alias of the table. The following statement shows an example:

```
select /*+ skewjoin(a) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1;
```

- Method 2: Specify the table name and possibly skewed columns in SKEWJOIN HINT. In the following statement, the c0 and c1 columns of Table a are skewed columns.

```
select /*+ skewjoin(a(c0, c1)) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1 and a.c2 = b.c2;
```

- Method 3: Specify the table name, columns, and skewed hot key values in SKEWJOIN HINT. If the skewed key values are of the STRING type, enclose each value with double quotation marks ("). In the following statement, (a.c0=1 and a.c1="2") and (a.c0=3 and a.c1="4") contain skewed hot key values.

```
select /*+ skewjoin(a(c0, c1)((1, "2"), (3, "4"))) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1 and a.c2 = b.c2;
```

Differences

- Method 3 in which the skewvalue parameter is specified is more efficient than Method 1 and Method 2 in which the skewvalue parameter is not specified.
- If you use Analyze and Freeride to collect TopK skewed hot key values, SKEWJOIN may automatically take effect without requiring you to specify a hint.

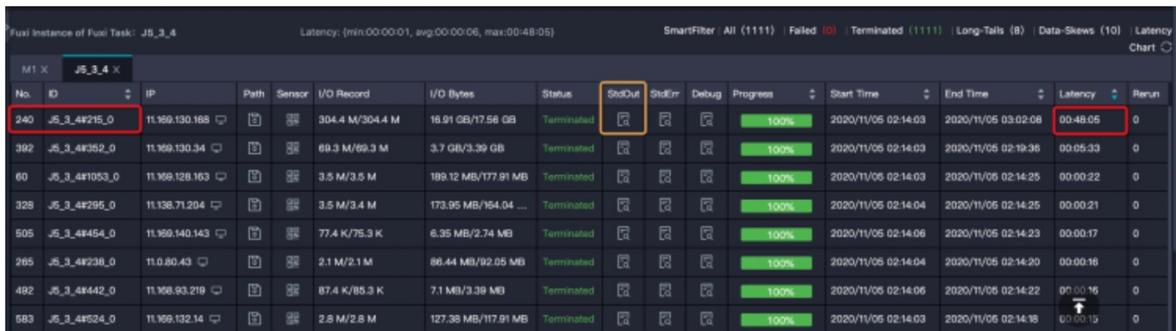
Examples

1. Identify the JOIN statement that causes data skew.

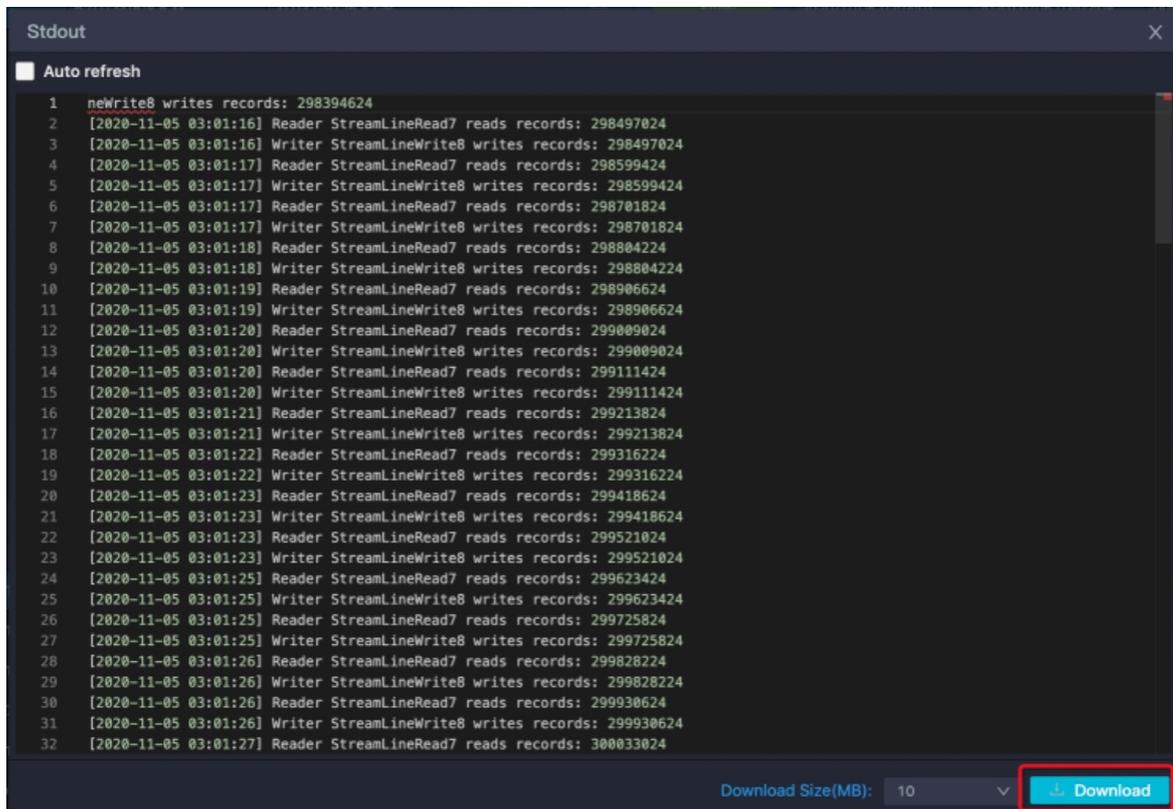
In the following screenshot captured on Logview, J5_3_4 is the Fuxi task that took the longest time to run.



Click the J5_3_4 task and query the instances of this task on the tab that appears. The query results show that the J5_3_4#215_0 instance takes the longest time to run and its I/O records and I/O bytes are much more than those of other instances.



This indicates that data skew occurs on the J5_3_4#215_0 instance. However, the JOIN statement that causes the data skew needs to be further determined. You can open the stdout of the skewed instance and the stdout of a non-skewed instance. The StdOut of the skewed instance is enclosed in the yellow rectangle in the preceding figure. The Stdout dialog box shown in the following figure appears. In most cases, the web page cannot display all the content of stdout. To view all the content, you can click **Download** in this dialog box.



The following figures show that the value of record count in StreamLineRead7 of the skewed instance is much greater than that of the non-skewed instance. Therefore, data skew occurs when data in StreamLineWrite7 and StreamLineRead7 is shuffled.

Skewed instance

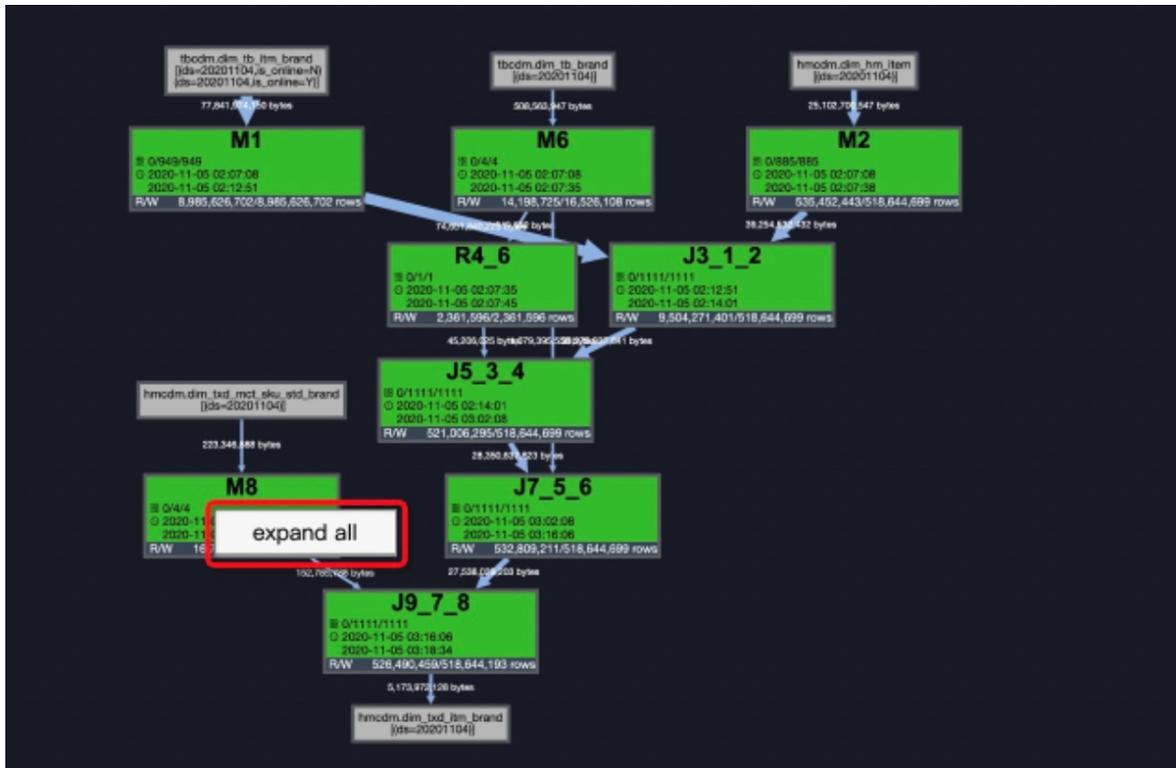
```
[2020-11-05 02:14:04] ----- Run Task: J5_3_4#215 -----
[2020-11-05 02:14:04] Reader StreamLineRead4, total estimated size: 68628, total record count: 2200
[2020-11-05 02:14:04] Reader StreamLineRead7, total estimated size: 5547886537, total record count: 184442043
[2020-11-05 02:14:06] Reader StreamLineRead4 reads records: 1824
[2020-11-05 02:14:06] Hash Join cursor MergeJoin2#hash#0 loaded small table: {
  "EqualGroupCount": 2200,
  "ExitStartTimeStamp": 1604513646,
  "HasBucketListDepth": 5,
  "HasRowCountInSingleGroup": 1,
  "MemoryUsedComplex(MB)": 0,
  "MemoryUsedFixed(MB)": 0,
  "MemoryUsedHashTable(MB)": 0,
  "MemoryUsedString(MB)": 0,
  "SmallTableInputCount": 2200} duration: 0
[2020-11-05 02:14:06] Reader StreamLineRead7 reads records: 1824
[2020-11-05 02:14:06] Writer StreamLineWrite8 writes records: 1824
[2020-11-05 02:14:06] Reader StreamLineRead7 reads records: 183424
```

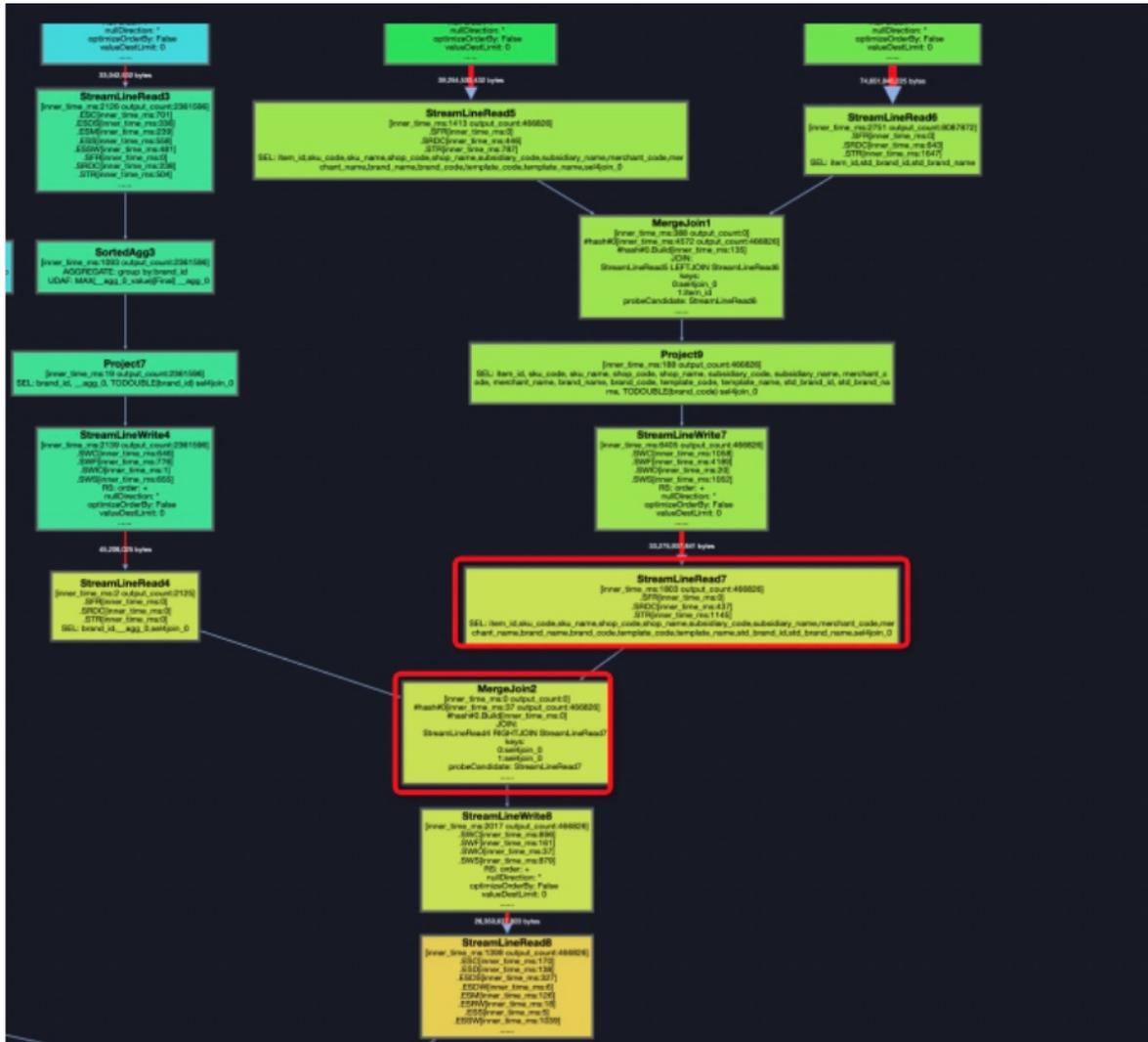
Non-skewed instance

```

[2020-11-05 02:14:04] ----- Run Task: J5_3_4#1053 -----
[2020-11-05 02:14:04] Reader StreamLineRead4, total estimated size: 69195, total record count: 2150
[2020-11-05 02:14:04] Reader StreamLineRead7, total estimated size: 605691502, total record count: 3534181
[2020-11-05 02:14:06] Hash Join cursor MergeJoin2#hash#0 loaded small table: {
  "EqualGroupCount": 2150,
  "InitStartTimestamp": 1604513646,
  "MaxBucketListDepth": 5,
  "MaxRowCountInSingleGroup": 1,
  "MemoryUsedComplex(MB)": 0,
  "MemoryUsedFixed(MB)": 0,
  "MemoryUsedHashTable(MB)": 0,
  "MemoryUsedString(MB)": 0,
  "SmallTableInputCount": 2150} duration: 0
[2020-11-05 02:14:06] Reader StreamLineRead7 reads records: 1024
[2020-11-05 02:14:06] Writer StreamLineWrite8 writes records: 1024
[2020-11-05 02:14:06] Reader StreamLineRead7 reads records: 103424
[2020-11-05 02:14:06] Writer StreamLineWrite8 writes records: 103424
[2020-11-05 02:14:07] Reader StreamLineRead7 reads records: 205824
[2020-11-05 02:14:07] Writer StreamLineWrite8 writes records: 205824
[2020-11-05 02:14:07] Reader StreamLineRead7 reads records: 308224
[2020-11-05 02:14:07] Writer StreamLineWrite8 writes records: 308224
  
```

On the directed acyclic graph (DAG), you can right-click the skewed instance and select **expand all** to find StreamLineWrite7 and StreamLineRead7.





The DAG shows that data skew occurs on StreamLineRead7 in MergeJoin2. MergeJoin2 is generated after the dim_hm_item and dim_tb_tm_brand tables are joined, and then the generated new table and the dim_tb_brand table are joined.



2. Locate the join based on the names of the related tables in Step 1. The result shows that data skew occurs in LEFT OUTER JOIN in the SQL statements and table t1 has data skew. To resolve this data skew issue, add the following command to the SQL statements:

```
/*+ skewjoin(t1)*/
```

```

29     t1.skf_code,
30     t1.skf_name,|
31     t1.brand_code,
32     t1.brand_name,
33     t2.std_brand_id,
34     t2.std_brand_name
35     from hmcdf.dim_hm_item t1
36     left outer join tbcdf.dim_tb_itm_brand t2
37     on cast(t1.item_id as string)=cast(t2.item_id as string)
38     and t2.ds=max_pt('tbcdf.dim_tb_itm_brand')
39     where t1.ds='20201104'
40     and t1.template_code in (
41         10002,
42         10006,
43         10007,
44         10018,
45         10019,
46         10021,
47         10022
48     )
49     ) t1
50     left outer join(
51         select brand_id as std_brand_id, max(brand_name) as std_brand_name
52         from tbcdf.dim_tb_brand
53         where ds=max_pt('tbcdf.dim_tb_brand')
54         and is_standard_brand='Y'
55         and cast(brand_id as bigint)>0
56         group by brand_id
57     ) t21
58     on t1.brand_code=t21.std_brand_id
59     left outer join(
60     select

```

Usage notes

- Supported join types: INNER JOIN allows you to specify a hint for either the right or left table of the JOIN operation. LEFT JOIN, SEMI JOIN, and ANTI JOIN allow you to specify a hint only for the left table. RIGHT JOIN allows you to specify a hint only for the right table. FULL JOIN does not support SKEWJOIN HINT.
- An Aggregate is run after SKEWJOIN HINT is added, which slows down the JOIN operation. Therefore, we recommend that you add SKEWJOIN HINT only to the JOIN statements that cause data skew.
- The data type of Left Side Join Key must be the same as that of Right Side Join Key for the JOIN statement to which SKEWJOIN HINT is added. If the data types are different, SKEWJOIN HINT becomes ineffective.
- After a hint is added, the optimizer runs an Aggregate to dynamically obtain the hot key values of the first 20 rows with the most duplicate key values. 20 is a default value. You can add the following flag parameter and specify the number of the rows whose hot key values you want to return.

```
set odps.optimizer.skew.join.topk.num = xx;
```

- You can specify a hint only for one of the two tables in the JOIN operation.
- In the JOIN statement to which SKEWJOIN HINT is added, left key = right key must be included. SKEWJOIN HINT cannot be added to a CARTESIAN JOIN statement.
- The following statement shows how to use SKEWJOIN HINT with other hints. Note that SKEWJOIN HINT cannot be added to a JOIN statement to which MAPJOIN HINT is added.
- On the Json Summary tab of Logview, you can search for the topk_agg field to check whether SKEWJOIN HINT takes effect. If such a field exists, SKEWJOIN HINT takes effect.

1.6.17.2. Insufficient computing resources

This topic describes the issue of insufficient computing resources and provides a solution.

Computing resources in MaxCompute may be insufficient due to improper planning and misuse of cluster resources.

Jobs with insufficient computing resources have the following common characteristics:

- The job output stops at a stage and the progress remains unchanged. For example, in the following figure, the progress of M1_Stg1 in the job remains at 0%. R2_1_Stg1 depends on M1_Stg1. Therefore, the progress of R2_1_Stg1 remains at 0% before M1_Stg1 is complete.

```
2016-01-29 13:52:09 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:14 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:19 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:24 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:29 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:34 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:39 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:44 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:49 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:54 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:52:59 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:04 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:09 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:15 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:20 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:25 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:30 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:35 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:40 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:45 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:50 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
2016-01-29 13:53:55 M1_Stg1_job0:0/0/5 [0%] R2_1_Stg1_job0:0/0/1 [0%]
```

- Task instances are in the Ready state on Logview, as shown in the following figure. This state indicates that resources are insufficient to run these task instances. After an instance obtains the necessary resources, its state changes to Running. If an instance is in the Waiting state, the instance waits for the completion of the task on which it depends.

	FuxiInstanceID	IP & Path	StdOut	StdErr	Status
1	Odps/odps_s...				Ready
2	Odps/odps_s...				Ready
3	Odps/odps_s...				Ready
4	Odps/odps_s...				Ready
5	Odps/odps_s...				Ready

Each job is split into multiple tasks based on an execution plan. These tasks form a directed acyclic graph (DAG). Multiple instances are concurrently invoked in each task to complete the computation. In most cases, the resources required to invoke an instance are a single-core CPU and 2 GB memory. To properly allocate resources, a quota group is assigned to each project. The quota group determines the upper limit of resources (CPU and memory) that are available for all jobs in the project. If the resources used by the jobs that concurrently run reach the upper limit of the quota group, the jobs stop running due to insufficient resources.

You can use one of the following methods to resolve this issue:

- Run jobs in off-peak hours.
- Contact O&M personnel to increase the resource capacity of the quota group for the project.

1.6.17.3. Methods to optimize storage in MaxCompute SQL

This topic describes the methods to optimize storage in MaxCompute SQL.

Reasonably configure partitioned tables

MaxCompute supports the concept of partitioning in a table. A partition refers to a partition space that is specified when a table is created. That is, a few fields in the table serve as partition key columns. In most cases, you can consider a partition as a directory in a file system. Each value of a partition key column is called a partition in MaxCompute. You can group multiple fields of a table to a single partition to create a multi-level partition. Multi-level partitions are similar to multi-level directories. If you specify the name of a partition you want to access, the system reads data only from that partition and does not scan the entire table. This reduces costs and improves efficiency.

Example of a partitioned table:

```
create table src (key string, value bigint) partitioned by (pt string);
select * from src where pt='20160901';
-- Specifies the partitioning format. MaxCompute takes only the data in the 20160901 partition as the input when MaxCompute generates a query plan.
```

Example of a non-partitioned table:

```
select * from src where key = 'MaxCompute';
-- The entire table is scanned in a query plan.
```

You can configure partitions by date or geographical region. You can also configure partitions based on your business requirements. Example:

```
create table if not exists sale_detail(
shop_name string,
customer_id string,
total_price double)
partitioned by (sale_date string,region string);
-- Create a two-level partitioned table, in which level-1 partitions are specified by sale_date, and level-2 partitions are specified by region.
```

Reasonably configure the table lifecycle

Storage resources in MaxCompute are valuable. You can configure the lifecycle of a table based on data usage. MaxCompute deletes data that exceeds the lifecycle threshold at the earliest opportunity to save storage space.

Example:

```
create table test3 (key boolean) partitioned by (pt string, ds string) lifecycle 100;
-- Create a table with a lifecycle of 100 days. If the last modification of the table or a partition occurred more than 100 days ago, MaxCompute deletes the table or partition.
```

 **Notice** The lifecycle takes a partition as the smallest unit. For a partitioned table, if some partitions reach the lifecycle threshold, they will be directly deleted. Partitions that have not reached the lifecycle threshold are not affected.

You can run the following command to change the lifecycle of a created table:

```
alter table table_name set lifecycle days;
```

Archive cold data

Some data needs to be retained permanently or for a long period of time. However, the frequency of accessing the data decreases over time. If the use frequency decreases to an extent, you can archive the data as RAID files. Data is not stored as three copies. The Cauchy Reed-Solomon algorithm is used to store data as six copies of the original data plus three parity blocks. This improves the effective storage ratio from 1:3 to 1:1.5. MaxCompute uses the bzip2 algorithm to archive tables with a higher data compression ratio than other algorithms. These two algorithms can be combined to save the storage space by more than 70%.

Format of an archiving command:

```
ALTER TABLE table_name [PARTITION(partition_name='partition_value')] ARCHIVE;
```

Example:

```
alter table my_log partition(ds='20170101') archive;
-- Archive the data with ds set to 20170101.
```

Merge small files

A large number of small files are generated during the Reduce computing process or during Tunnel-based real-time data collection. This causes the following issues:

- More instance resources are required because the number of files a single instance can process is limited. This results in a waste of resources and affects the overall execution performance.
- The workload on the file system is high, and the disk utilization is affected.

You can use one of the following methods to merge small files:

- ALTER command: Run the following command in the console command line to merge small files:

```
ALTER TABLE tablename [PARTITION] MERGE SMALLFILES;
```

- SQL statement: After SQL statements are executed, set `odps.task.merge.enabled` to true and run a Fuxi job to merge small files.

1.6.17.4. UDF OOM

This topic describes the UDF OOM issue and provides a solution.

An OOM error is reported when some jobs are running. Detailed error information:

```
FAILED: ODPS-0123144: Fuxi job failed - WorkerRestart errCode:9,errMsg:SigKill(OOM), usually caused by OOM(out of memory)
```

You can configure UDF runtime parameters to fix the error. Example:

```
odps.sql.mapper.memory=3072;
set odps.sql.udf.jvm.memory=2048;
set odps.sql.udf.python.memory=1536;
```

1.6.18. Limits of MaxCompute SQL statements

This topic describes all the limits of MaxCompute SQL statements.

Item	Maximum value/Limit	Category	Description
Table name length	128 bytes	Length	A table or column name can contain only letters, digits, and underscores (_). It must start with a letter.
Comment length	1,024 bytes	Length	A comment is a valid string that cannot exceed 1,024 bytes in length.
Column definitions in a table	1,200	Quantity	A table can contain a maximum of 1,200 column definitions.
Partitions in a table	60,000	Quantity	A table can contain a maximum of 60,000 partitions.
Partition levels of a table	6	Quantity	A table can contain a maximum of six levels of partitions.
<code>INSERT</code> destination objects	256	Quantity	In a <code>MULTI-INSERT</code> operation, you can insert data into a maximum of 256 tables at a time.
<code>UNION ALL</code>	256	Quantity	A <code>UNION ALL</code> operation can be performed on a maximum of 256 tables.

Item	Maximum value/Limit	Category	Description
MAPJOIN	128	Quantity	A MAPJOIN operation can be performed on a maximum of 128 small tables.
MAPJOIN memory	512 MB	Size	The memory size for all small tables on which a MAPJOIN operation is performed cannot exceed 512 MB.
Window functions	5	Quantity	A SELECT statement can contain a maximum of five window functions.
ptinsubq	1,000	Quantity	A PT IN SUBQUERY statement can generate a maximum of 1,000 rows.
Length of an SQL statement	2 MB	Length	An SQL statement cannot exceed 2 MB in length. This limit is suitable for the scenarios where you use an SDK to call SQL statements.
Conditions of a WHERE clause	256	Quantity	A WHERE clause can contain a maximum of 256 conditions.
Length of a column record	8 MB	Length	A column record in a table cannot exceed 8 MB in length.
Parameters in an IN clause	1024	Quantity	This item specifies the maximum number of parameters in an IN clause, for example, in (1,2,3...,1024) . Excessive parameters will affect the compilation performance. We recommend that you use no more than 1,024 parameters, but this is not a fixed upper limit.
jobconf.json	1 MB	Size	The maximum size of the jobconf.json file is 1 MB. If a table contains a large number of partitions, the size of the jobconf.json file may exceed 1 MB.
View	Not writable	Operation	A view is not writable and does not support the INSERT operation.
Data type and position of a column	Unmodifiable	Operation	The data type and position of a column cannot be modified.
Java UDFs	Not allowed to be abstract or static	Operation	Java UDFs cannot be abstract or static .
Partitions that can be queried	10,000	Quantity	A maximum of 10,000 partitions can be queried.

Item	Maximum value/Limit	Category	Description
SQL execution plans	1 MB	Size	The size of an execution plan generated from MaxCompute SQL statements cannot exceed 1 MB. Otherwise, the error message <code>FAIL FD: ODPS-0010000:System internal error - The Size of Plan is too large</code> is reported.

 **Notice** The preceding limits cannot be manually modified.

1.6.19. Appendix

1.6.19.1. Escape characters

This topic describes the escape characters in MaxCompute SQL.

In MaxCompute SQL, the backslash (\) is an escape character that invokes an alternative representation on the special characters in a character string. This escape character may also indicate a literal interpretation of the characters that follow the escape character. If the backslash (\) in a string constant is followed by three valid octal digits in the range from 001 to 177, the system converts the ASCII values to the corresponding characters. The following table describes the mapping between escape characters and represented special characters.

Escape character	Represented special character
\b	Backspace
\t	Tab
\n	Newline
\r	Carriage-return
\'	Single quotation mark
\"	Double quotation marks
\\	Backslash
\;	Semicolon
\Z	Control-Z
\0 or \00	Ending character

Examples:

- Use an escape character to indicate a literal interpretation of the characters that follow the escape character.

String constants in MaxCompute SQL can be expressed in single quotation marks (' ') or double quotation marks (" "). You can include double quotation marks (" ") in a string that is enclosed in single quotation marks (' ') or include single quotation marks (' ') in a string that is enclosed in double quotation marks (" "). Examples:

```
"I'm a happy manong."  
'\|'m a happy manong.'
```

- Use an escape character to represent special characters. Examples:

```
select length('a\tb');  
-- 3 is displayed in the returned result. This indicates that the string contains three characters and that \t is treated as one character.
```

```
select 'a\ab',length('a\ab');  
-- aab, 3 is displayed in the returned result. \a is treated as Letter a.
```

1.6.19.2. LIKE usage

This topic describes how to use the LIKE operator for character matching.

In LIKE character matching, a percent sign (%) is a wildcard character that matches an arbitrary number of characters. An underscore (_) is a wildcard character that matches a single character.

To match percent signs (%) or underscores (_), escape them with double backslashes (\\). Specifically, \\% or _ is used for the match.

```
'abcd' like 'ab%'  
-- True.  
'abcd' like 'ab_'  
-- False.  
'ab_cde' like 'ab\\_c%';  
-- True.
```

 **Note** MaxCompute SQL statements support only UTF-8 character sets. If data is encoded in another format, the calculation result may be incorrect.

1.6.19.3. Regular expressions

This topic describes regular expressions in MaxCompute SQL.

Regular expressions in MaxCompute SQL use the Perl Compatible Regular Expressions (PCRE) standards and are matched by character.

RLIKE

You can use the RLIKE statement to match regular expressions. The following table lists the supported metacharacters.

Metacharacter	Description
^	Match the beginning of a line.
\$	Match the end of a line.

Metacharacter	Description
.	Match any characters.
*	Match zero or more instances of the preceding character or character pattern.
+	Match one or more instances of the preceding character or character pattern.
?	Match zero or one instance of the preceding character or character pattern.
?	Match a modifier. If this character follows one of other delimiters (*, +, ?, {n}, {n,}, or {n,m}), the match pattern is non-greedy. In the non-greedy pattern, as few characters as possible are matched with the searched character string. In the default greedy pattern, as many characters as possible are matched with the searched character string.
A B	Match A or B.
(abc)*	Match zero or more instances of the abc sequence.
{n} or {m,n}	The number of matches.
[ab]	Match any character (a or b) in the brackets. Fuzzy match is supported.
[a-d]	Match any of the following characters: a, b, c, and d.
[^ab]	Match any character that is not a or b. ^ indicates 'non'.
[::]	For more information, see the following table.
\	The escape character.
\n	n indicates a digit from 1 to 9 and is backward referenced.
\d	Digits.
\D	Non-digit characters.

POSIX character groups

Character group	Description	Valid value
[[:alnum:]]	Letters and digits	[a-zA-Z0-9]
[[:alpha:]]	Letters	[a-zA-Z]
[[:ascii:]]	ASCII characters	[\x00-\x7F]
[[:blank:]]	Spaces and tab characters	[\t]
[[:cntrl:]]	Control characters	[\x00-\x1F\x7F]
[[:digit:]]	Digits	[0-9]
[[:graph:]]	Characters other than whitespace characters	[\x21-\x7E]

Character group	Description	Valid value
[:lower:]	Lowercase letters	[a-z]
[:print:]	[:graph:] and whitespace characters	[\x20-\x7E]
[:punct:]	Punctuation marks	[!\"#\$%&'()*+,-./:;<=>?@\^_`{ }~]
[:space:]	Whitespace characters	[\t\r\n\v\f]
[:upper:]	Uppercase letters	[A-Z]
[:xdigit:]	Hexadecimal characters	[A-Fa-f0-9]

Escape characters

The system uses a backslash (\) as the escape character. A backslash (\) in a regular expression indicates a second escape. For example, a regular expression is used to match the `a+b` string. The plus sign (+) is a special character in the expression and must be escaped. In the regular expression engine, the string is expressed as `a\\+b`. The system must perform an escape on the expression. Therefore, the expression that can match the string is `a\\\\+b`.

Assume that the `test_dual` table exists. Example:

```
select 'a+b' rlike 'a\\\\+b' from test_dual;
```

Returned result:

```
+-----+
|_c1 |
+-----+
| true |
+-----+
```

The `\` character is a special character in the regular expression engine. To match this character, it is expressed as `\\` in the engine. Then, the system performs an escape on the expression. As a result, the `\` character is expressed as `\\`.

```
select 'a\\b', 'a\\b' rlike 'a\\\\b' from test_dual;
```

Returned result:

```
+-----+-----+
|_c0|_c1 |
+-----+-----+
| a\b | false |
+-----+-----+
```

Note If a MaxCompute SQL statement includes `a\\b`, `a\b` is displayed in the output because MaxCompute escapes the expression.

If a string includes tab characters, the system reads `\t` and stores it as one character. Therefore, it is a common character in regular expressions.

```
select 'a\tb', 'a\tb' rlike 'a\tb' from test_dual;
```

Returned result:

```
+-----+-----+
|_c0  |_c1 |
+-----+-----+
|a  b|true|
+-----+-----+
```

1.6.19.4. Reserved words and keywords

This topic describes all reserved words and keywords in MaxCompute SQL.

Notice

- When you name a table, a column, or a partition, do not use reserved words or keywords. Otherwise, an error may occur.
- Reserved words are not case-sensitive.

```
% & && ( ) * +
- . / ; < <= <>
= > >= ? ADD AFTER ALL
ALTER ANALYZE AND ARCHIVE ARRAY AS ASC
BEFORE BETWEEN BIGINT BINARY BLOB BOOLEAN BOTH DECIMAL
BUCKET BUCKETS BY CASCADE CASE CAST CFILE
CHANGE CLUSTER CLUSTERED CLUSTERSTATUS COLLECTION COLUMN COLUMNS
COMMENT COMPUTE CONCATENATE CONTINUE CREATE CROSS CURRENT
CURSOR DATA DATABASE DATABASES DATE DATETIME DBPROPERTIES
DEFERRED DELETE DELIMITED DESC DESCRIBE DIRECTORY DISABLE
DISTINCT DISTRIBUTE DOUBLE DROP ELSE ENABLE END EXCEPT
ESCAPED EXCLUSIVE EXISTS EXPLAIN EXPORT EXTENDED EXTERNAL
FALSE FETCH FIELDS FILEFORMAT FIRST FLOAT FOLLOWING
FORMAT FORMATTED FROM FULL FUNCTION FUNCTIONS GRANT
GROUP HAVING HOLD_DDLTIME IDXPROPERTIES IF IMPORT IN
INDEX INDEXES INPATH INPUTDRIVER INPUTFORMAT INSERT INT
INTERSECT INTO IS ITEMS JOIN KEYS LATERAL
LEFT LIFECYCLE LIKE LIMIT LINES LOAD LOCAL
LOCATION LOCK LOCKS LONG MAP MAPJOIN MATERIALIZED
MINUS MSCK NOT NO_DROP NULL OF OFFLINE OFFSET
ON OPTION OR ORDER OUT OUTER OUTPUTDRIVER
OUTPUTFORMAT OVER OVERWRITE PARTITION PARTITIONED PARTITIONPROPERTIES PARTITIONS
PERCENT PLUS PRECEDING PRESERVE PROCEDURE PURGE RANGE
RCFILE READ READONLY READS REBUILD RECORDREADER RECORDWRITER
REDUCE REGEXP RENAME REPAIR REPLACE RESTRICT REVOKE
RIGHT RLIKE ROW ROWS SCHEMA SCHEMAS SELECT
SEMI SEQUENCEFILE SERDE SERDEPROPERTIES SET SHARED SHOW
SHOW_DATABASE SMALLINT SORT SORTED SSL STATISTICS STATUS STORED
STREAMTABLE STRING STRUCT TABLE TABLESAMPLE TBLPROPERTIES
TEMPORARY TERMINATED TEXTFILE THEN TIMESTAMP TINYINT TO
TOUCH TRANSFORM TRIGGER TRUE TYPE UNARCHIVE UNBOUNDED UNDO
UNION UNIONTYPE UNIQUEJOIN UNLOCK UNSIGNED UPDATE USE
USING UTC UTC_TIMESTAMP VIEW WHEN WHERE WHILE DIV
```

1.6.19.5. Settings of new data types

This topic describes the precautions for configuring new data types in MaxCompute SQL.

If you want to read a table that includes new data types, you are not required to add the `setodps.sql.type.system.odps2=true;` flag. However, you must take note of the following points:

- If the flag is not added, the read data is implicitly converted into the original data type for all computations.
- If the flag is not added for integer constants, the `BIGINT` type is used, and an error message is reported.
- If you write data to a table and the data is in passthrough mode, you can choose not to add the new data type flag. However, if you want to calculate the data, an error is reported because the implicit data type conversion is invalid.

1.6.19.6. ACID semantics of MaxCompute concurrent write jobs

This topic describes the ACID semantics of MaxCompute concurrent write jobs.

Terms

- Operation: a single job submitted in MaxCompute.
- Data object: an object that contains data, such as a non-partitioned table or a partition.
- INTO job: an SQL job that contains the `INTO` keyword, such as `INSERT INTO` or `DYNAMIC INSERT INTO`.
- OVERWRITE job: an SQL job that contains the `OVERWRITE` keyword, such as `INSERT OVERWRITE` or `DYNAMIC INSERT OVERWRITE`.
- Data upload by using Tunnel: an `INTO` or `OVERWRITE` job.

Description of ACID semantics

- Atomicity: An operation is either fully complete or not performed at all. That is, an operation is not partially performed.
- Consistency: The integrity of data objects is maintained when an operation is performed.
- Isolation: An operation can be performed without depending on other concurrent operations.
- Durability: After an operation is complete, modified data is permanently valid and is not lost even if a system failure occurs.

Scenarios of ACID semantics for MaxCompute

- Atomicity
 - When multiple jobs conflict with each other, MaxCompute ensures that only one job succeeds.
 - The atomicity of the `CREATE`, `OVERWRITE`, and `DROP` operations on a single table or partition is ensured.
 - The atomicity of cross-table operations such as `MULTI-INSERT` cannot be ensured.
 - In extreme cases, the following operations may not be atomic:
 - A `DYNAMIC INSERT OVERWRITE` operation that is performed on more than 10,000 partitions.
 - An `INTO` operation. `INTO` operations cannot ensure atomicity because data cleansing fails during a transaction rollback. However, the transaction rollback does not cause raw data loss.
- Consistency
 - `OVERWRITE` jobs ensure consistency.
 - If an `INTO` job fails due to a conflict, data from the failed job may remain.

- Isolation
 - Non-INTO operations ensure that read operations are submitted.
 - In INTO operations, some read operations may not be submitted.
- Durability
 - MaxCompute ensures data durability.

ACID semantics of transactional tables

- Based on existing ACID semantics, INTO operations ensure that read operations are submitted and no data remains after an operation fails due to a conflict.
- The atomicity of the UPDATE and DELETE operations on a single non-partitioned table or a partition is ensured. For example, if two update jobs concurrently run to modify the data of a partition, only one job succeeds. The partition is not partially updated by one job. It is also not updated by both jobs with part of the updated data remained.

Conflicts among concurrent operations: When two jobs concurrently run and write data to the same destination table, a conflict may occur. In the event of a conflict, the job that ends earlier will succeed, and the job that ends later will fail due to the conflict.

The following table describes the conflicts that may occur when two jobs concurrently run and write data to the same non-partitioned table or partition. The first column lists the job that ends earlier. The other columns list the jobs that end later.

Before/After	INSERT OVERWRITE	INSERT INTO	UPDATE/DELETE	MERGE SMALLFILES
INSERT OVERWRITE	Both jobs succeed. The result data of the INSERT OVERWRITE operation that ends later will overwrite that of the INSERT OVERWRITE operation that ends earlier.	Both jobs succeed. The result data of the INSERT INTO operation is appended to that of the INSERT OVERWRITE operation.	The INSERT OVERWRITE operation modifies the data of the non-partitioned table or partition. An error is reported in the UPDATE operation.	The INSERT OVERWRITE operation modifies the data of the non-partitioned table or partition. An error is reported in the MERGE SMALLFILES operation.
INSERT INTO	Both jobs succeed. The result data of the INSERT OVERWRITE operation overwrites that of the INSERT INTO operation.	Both jobs succeed. The result data of the two INSERT INTO operations is merged.	The INSERT INTO operation modifies the data of the non-partitioned table or partition. An error is reported in the UPDATE operation.	The INSERT INTO operation modifies the data of the non-partitioned table or partition. An error is reported in the MERGE SMALLFILES operation.
UPDATE/DELETE	Both jobs succeed. The result data of the INSERT OVERWRITE operation overwrites that of the UPDATE operation.	Both jobs succeed. The result data of the INSERT INTO operation is appended to that of the UPDATE operation.	The UPDATE operation that ends earlier modifies the data of the non-partitioned table or partition. An error is reported in the UPDATE operation that ends later.	The UPDATE operation modifies the data of the non-partitioned table or partition. An error is reported in the MERGE SMALLFILES operation.

Before/After	INSERT OVERWRITE	INSERT INTO	UPDATE/DELETE	MERGE SMALLFILES
MERGE SMALLFILES	Both jobs succeed. The result data of the INSERT OVERWRITE operation overwrites that of the MERGE SMALLFILES operation.	Both jobs succeed. The result data of the INSERT INTO operation is appended to that of the MERGE SMALLFILES operation.	The MERGE SMALLFILES operation modifies the data of the non-partitioned table or partition. An error is reported in the UPDATE operation.	The MERGE SMALLFILES operation that ends earlier modifies the data of the non-partitioned table or partition. An error is reported in the MERGE SMALLFILES operation that ends later.

The following table simplifies the preceding rules and describes only whether an error is reported for an operation after data changes.

Operation	Error reported upon data changes
INSERT OVERWRITE	N
INSERT INTO	N
UPDATE/DELETE	Y
MERGE SMALLFILES	Y

No conflict is generated for INSERT operations in the event of data changes. An error is reported in the UPDATE/DELETE or MERGE SMALLFILES operation if the destination non-partitioned table or partition has data changes.

 **Note**

- MINOR/MAJOR COMPACT is classified as MERGE SMALLFILES operations.
- In extreme scenarios, if multiple jobs are all in the metadata update phase, an error may be reported due to a concurrency conflict among these update operations.

1.7. MaxCompute Tunnel

1.7.1. Overview

This topic describes how to upload data to MaxCompute or download data from MaxCompute. This topic also describes related limits.

MaxCompute provides two types of channels for data uploads and downloads:

- DataHub: This channel is used to upload or download data in real time. It includes the OGG, Flume, Logstash, and Fluentd plug-ins.
- Tunnel: This channel is used to upload or download large amounts of data at a time. It includes the MaxCompute client, DataWorks, DTS, Sqoop, Kettle plug-in, and MaxCompute Migration Assist (MMA).

DataHub and Tunnel provide their own SDKs. A variety of data upload and download tools are derived from the SDKs.

The tools can meet the data upload and download requirements of most common scenarios in which data is migrated to the cloud.

Tunnel service connections

DataHub and Tunnel use different endpoints in different network environments. You must also select different endpoints to connect to the service.

Limits

- Limits on Tunnel-based data uploads:
 - You cannot run Tunnel commands to upload or download data of the ARRAY, MAP, or STRUCT type.
 - No limits are specified for the upload speed. The upload speed depends on the network bandwidth and server performance.
 - The number of retries is limited. If the number of retries exceeds the limit, the next block is uploaded. After data is uploaded, you can execute the `select count(*) from table_name` statement to check whether any data is lost.
 - By default, a project supports a maximum of 2,000 concurrent Tunnel connections.
 - On the server, the lifecycle of a session is 24 hours. A session can be shared among processes and threads on the server, but you must make sure that each block ID is unique.
 - MaxCompute ensures the validity of concurrent writes based on atomicity, consistency, isolation, durability (ACID).

- Limits on DataHub-based data uploads:
 - The size of each field cannot exceed its upper limit.

 **Note** The size of a string cannot exceed 8 MB.

- During an upload, multiple data records are packaged.
- Limits on TableTunnel SDK interfaces:
 - A block ID must be greater than or equal to 0 but less than 20,000. The size of the data that you want to upload in a block cannot exceed 100 GB.
 - The lifecycle of a session is 24 hours. If you want to transfer large amounts of data, more than 24 hours are required. In this case, we recommend that you transfer the data in multiple sessions.
 - The lifecycle of an HTTP request that corresponds to a RecordWriter is 120 seconds. If no data flows over an HTTP connection within 120 seconds, the server closes the connection.

1.7.2. Selection of tools to migrate data to the cloud

MaxCompute provides a variety of data upload and download tools, which can be used in different scenarios to migrate data to the cloud. This topic describes the selection of data transmission tools in three typical scenarios.

Hadoop data migration

You can use MaxCompute Migration Assist (MMA), Sqoop, and DataWorks to migrate Hadoop data.

- If you use DataWorks, DataX is required.
- If you use Sqoop, a MapReduce job runs on the original Hadoop cluster for distributed data transmission to MaxCompute.
- If you use MMA, Meta Carrier is required to access your Hive metastore service and capture Hive metadata. Then, MMA generates DDL statements that are used to create MaxCompute tables and partitions, as well as Hive UDTF SQL statements for data migration.

Synchronization of data in a database

To synchronize data from a database to MaxCompute, you must select a tool based on the database type and synchronization policy.

- Use DataWorks for offline batch synchronization. DataWorks supports a wide range of database types, such as MySQL, SQL Server, and PostgreSQL.
- Use the OGG plug-in for real-time synchronization of data in an Oracle database.
- Use DTS for real-time synchronization of data in an ApsaraDB RDS database.

Log collection

You can use tools such as Flume, Fluentd, and Logstash to collect logs.

1.7.3. Introduction to the tools

MaxCompute supports a wide range of data upload and download tools. The source code for most of the tools can be found and maintained on the open source community GitHub. You can select the appropriate tools to upload and download data based on usage scenarios. This topic describes these tools.

Alibaba Cloud services

- Data Integration of DataWorks (Tunnel)

Data Integration of DataWorks is a stable, efficient, and scalable data synchronization platform. It is designed to provide full offline and incremental real-time data synchronization, integration, and exchange services for the heterogeneous data storage systems on Alibaba Cloud.

Data synchronization tasks support the following data sources: MaxCompute, ApsaraDB RDS (MySQL, SQL Server, and PostgreSQL), Oracle, FTP, AnalyticDB, OSS, ApsaraDB for Memcache, and PolarDB-X.

- MaxCompute client (Tunnel)

Based on the batch data tunnel SDK, the client provides built-in Tunnel commands for data uploads and downloads.

- DTS (Tunnel)

Data Transmission Service (DTS) is an Alibaba Cloud data service that supports data exchange among multiple data sources, such as Relational Database Management System (RDBMS), NoSQL, and Online Analytical Processing (OLAP). It provides data transmission features, such as data migration, real-time data subscription, and real-time data synchronization.

DTS supports data synchronization from ApsaraDB RDS and MySQL instances to MaxCompute tables. Other data sources are not supported.

Open source products

The projects that correspond to each product are open-sourced. You can visit [aliyun-maxcompute-data-collectors](#) to view details.

- Sqoop (Tunnel)

Sqoop 1.4.6 in the community is further developed to provide enhanced MaxCompute support. It imports data from relational databases such as MySQL and data from HDFS or Hive to MaxCompute tables. It also exports data from MaxCompute tables to relational databases such as MySQL.

- Kettle (Tunnel)

Kettle is an open source extract, transform, load (ETL) tool that is developed in Java. It runs on Windows, UNIX, or Linux, and provides graphic interfaces for you to define the data transmission topology by using drag-and-drop components.

- Apache Flume (DataHub)
 - Apache Flume is a distributed and reliable system. It efficiently collects large volumes of log data from different data sources and then aggregates and stores the data in a centralized data storage. It supports multiple Source and Sink plug-ins.
 - The DataHub Sink plug-in of Apache Flume allows you to upload log data to DataHub in real time and archive the data in MaxCompute tables.
- Fluentd (DataHub)
 - Fluentd is an open source software product. It collects logs, such as application logs, system logs, and access logs, from various sources. It allows you to use plug-ins to filter log data and store the data in different data processors, including MySQL, Oracle, MongoDB, Hadoop, and Treasure Data.
 - The DataHub plug-in of Fluentd allows you to upload log data to DataHub in real time and archive the data in MaxCompute tables.
- Logstash (DataHub)
 - Logstash is an open source log collection and processing framework. The logstash-output-datahub plug-in allows you to import data to DataHub. This tool can be easily configured to collect and transmit data. It can be used together with MaxCompute or StreamCompute to easily create an all-in-one streaming data solution from data collection to analytics.
 - The DataHub plug-in of Logstash allows you to upload log data to DataHub in real time and archive the data in MaxCompute tables.
- OGG (DataHub)

The DataHub plug-in of OGG allows you to incrementally synchronize data in an Oracle database to DataHub in real time and archive the data in MaxCompute tables.
- MMA

MaxCompute Migration Assist (MMA) uses Meta Carrier to access your Hive metastore service and capture Hive metadata. It then generates DDL statements that are used to create MaxCompute tables and partitions, as well as Hive UDTF SQL statements for data migration.

1.7.4. Tunnel SDK overview

1.7.4.1. Overview

Data upload and download tools provided by MaxCompute are developed based on Tunnel SDK. This topic describes the major APIs of Tunnel SDK.

The usage of the SDK varies based on its version. For more information, see [SDK Java Doc](#).

API	Description
TableTunnel	The entry class that is used to access MaxCompute Tunnel.
TableTunnel.UploadSession	A session that uploads data to a MaxCompute table.
TableTunnel.DownloadSession	A session that downloads data from a MaxCompute table.
InstanceTunnel	The entry class that is used to access MaxCompute Tunnel.
InstanceTunnel.DownloadSession	A session that downloads data from a MaxCompute SQL instance. The SQL instance must start with the SELECT keyword and is used to query data.

Note

- If you use Maven, you can search for odps-sdk-core in the Maven repository to find the latest version of the SDK for Java. You can configure the Maven dependency in the following way:

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>odps-sdk-core</artifactId>
  <version>0.36.2</version>
</dependency>
```

- The endpoint of MaxCompute Tunnel supports automatic routing based on the MaxCompute endpoint settings.

1.7.4.2. TableTunnel

This topic describes the TableTunnel API.

TableTunnel is an entry class of the MaxCompute Tunnel service. You can use TableTunnel to upload or download only table data. Views cannot be uploaded or downloaded.

Definition

The following code defines the TableTunnel API.

```
public class TableTunnel {
  public DownloadSession createDownloadSession(String projectName, String tableName);
  public DownloadSession createDownloadSession(String projectName, String tableName, PartitionSpec partitionSpec);
  public UploadSession createUploadSession(String projectName, String tableName);
  public UploadSession createUploadSession(String projectName, String tableName, PartitionSpec partitionSpec);
  public DownloadSession getDownloadSession(String projectName, String tableName, PartitionSpec partitionSpec, String id);
  public DownloadSession getDownloadSession(String projectName, String tableName, String id);
  public UploadSession getUploadSession(String projectName, String tableName, PartitionSpec partitionSpec, String id);
  public UploadSession getUploadSession(String projectName, String tableName, String id); public void setEndpoint(String endpoint);
}
```

Description

- The lifecycle of a TableTunnel instance starts from the time it is created to the time data upload or download is complete.
- TableTunnel provides a method to create UploadSession and DownloadSession objects. TableTunnel.UploadSession is used to upload data. TableTunnel.DownloadSession is used to download data.
- A session refers to the process of uploading or downloading a table or partition. A session consists of one or more HTTP requests to Tunnel RESTful APIs.
- In an upload session, each RecordWriter matches an HTTP request and is identified by a unique block ID. The block ID is the name of the file that corresponds to the RecordWriter.
- If you use the same block ID to enable a RecordWriter multiple times in the same session, the data uploaded after the RecordWriter calls the close() method for the last time overwrites all the data that is previously uploaded. This feature can be used to retransmit a data block that fails to be uploaded.

- In UploadSession of TableTunnel:
 - If the boolean overwrite parameter is not specified, the INSERT INTO statement is used.
 - If the boolean overwrite parameter is set to True, the INSERT OVERWRITE statement is used.
 - If the boolean overwrite parameter is set to False, the INSERT INTO statement is used.

Descriptions of INSERT OVERWRITE and INSERT INTO:

- INSERT INTO: Upload sessions of the same table or partition do not affect each other. Data uploaded in each session is saved in different directories.
- INSERT OVERWRITE: All data in a table or partition is overwritten by the data in the current upload session. If you use this statement to upload data, do not perform concurrent operations on the same table or partition.

Implementation process

1. The RecordWriter.write() method uploads your data as files to a temporary directory.
2. The RecordWriter.close() method moves the files from the temporary directory to a data directory.
3. The session.commit() method moves all files from the data directory to the directory in which the required table is saved, and updates the table metadata. This way, the data moved to a table in the current job is visible to other MaxCompute jobs such as SQL and MapReduce jobs.

Limits

- The value of a block ID must be greater than or equal to 0 but less than 20000. The size of the data that can be uploaded in a block cannot exceed 100 GB.
- A session is uniquely identified by its ID. The lifecycle of a session is 24 hours. If your session times out because large amounts of data are transmitted, you must transmit your data in multiple sessions.
- The lifecycle of an HTTP request that corresponds to a RecordWriter is 120 seconds. If no data flows over an HTTP connection within 120 seconds, the server closes the connection.

 **Note** HTTP has an 8 KB buffer. When you call the RecordWriter.write() method, your data may be saved to the buffer and no inbound traffic flows over the HTTP connection. In this case, you can call the TunnelRecordWriter.flush() method to forcibly flush data out of the buffer.

- If you use a RecordWriter to write logs to MaxCompute, the write operation may time out due to unexpected traffic fluctuations. To avoid such issues, take note of the following points:
 - We recommend that you do not use a RecordWriter for each data record. If you use a RecordWriter for each data record, a large number of small files are generated, because each RecordWriter corresponds to a file. This affects the performance of MaxCompute.
 - If the size of cached code reaches 64 MB, we recommend that you use a RecordWriter to write multiple data records at the same time.
- The lifecycle of a RecordReader is 300 seconds.

1.7.4.3. InstanceTunnel

This topic describes the InstanceTunnel API.

InstanceTunnel is an entry class to access the MaxCompute Tunnel service. You can use InstanceTunnel to download the execution results of an SQL instance that executes a SELECT statement.

Definition

The following code shows the definition of the InstanceTunnel API:

```
public class InstanceTunnel{
    public DownloadSession createDownloadSession(String projectName, String instanceID);
    public DownloadSession createDownloadSession(String projectName, String instanceID, boolean limitEnabled);
    public DownloadSession getDownloadSession(String projectName, String id);
}
```

Parameters:

- projectName: the name of a project.
- instanceID: the ID of an instance.

Limits

InstanceTunnel provides an easy way to obtain instance execution results. However, it is subject to the following permission limits to ensure data security:

- If the number of data records is less than or equal to 10,000, all users who have read permissions on the specified instance can use InstanceTunnel to download the data records. The same rule applies to data queries by calling a RESTful API.
- If the number of data records is greater than 10,000, only users who have the read permissions on all the source tables from which the specified instance queries data can use InstanceTunnel to download the data records.

1.7.4.4. UploadSession

This topic describes the UploadSession API.

This API is used to upload data to MaxCompute tables.

Definition

The following code defines the UploadSession API:

```
public class UploadSession {
    UploadSession(Configuration conf, String projectName, String tableName,
        String partitionSpec) throws TunnelException;
    UploadSession(Configuration conf, String projectName, String tableName,
        String partitionSpec, String uploadId) throws TunnelException;
    public void commit(Long[] blocks);
    public Long[] getBlockList();
    public String getId();
    public TableSchema getSchema();
    public UploadSession.Status getStatus();
    public Record newRecord();
    public RecordWriter openRecordWriter(long blockId);
    public RecordWriter openRecordWriter(long blockId, boolean compress);
    public RecordWriter openBufferedWriter();
    public RecordWriter openBufferedWriter(boolean compress);
}
```

 Notice

- Block IDs that are used within the same upload session must be unique. After you use a block ID to enable RecordWriter, write multiple data records at the same time, call close, and then call commit to complete data upload in an upload session, you cannot use this block ID to enable another RecordWriter to write data.
- The maximum size of a block is 100 GB. We recommend that the volume of data written to each block be greater than 64 MB. Otherwise, the computing performance deteriorates significantly.
- The lifecycle of a session on the server is 24 hours.
- When you upload data, a network action is triggered each time RecordWriter writes 8 KB of data. If no network actions are triggered within 120 seconds, the server closes the connection and RecordWriter becomes unavailable. You must enable a new RecordWriter to write data.
- We recommend that you use the openBufferedWriter operation to upload data. This operation does not show the blockId value but contains an internal data cache. If a block fails to be uploaded, this operation automatically retries the upload process.
- The overwrite mode is added by using the commit method. You can use the overwrite mode to submit data. If you submit data in overwrite mode, the submitted data overwrites the existing data in the table or partition.

 **Notice** Undefined behavior occurs when you submit data in overwrite mode in multiple concurrent sessions. This may affect data accuracy. To avoid this issue, you must determine the number of concurrent sessions in which you submit data in overwrite mode.

Description

- Lifecycle: indicates the period from the time an upload instance is created to the time data is uploaded.
- Upload instance. You can call the Construct or method or use TableTunnel to create an upload instance.
 - Request mode: synchronous.
 - The server creates a session for the upload instance and generates a unique upload ID to identify the upload instance. You can run the `getId` command on the client to obtain the upload ID.
- Data upload
 - Request mode: synchronous.
 - Call the openBufferedWriter operation to generate a RecordWriter instance. The blockId parameter identifies the data that is uploaded this time and describes the data position in the whole table. The value of blockId is in the range of [0,20000]. If the upload fails, you can upload the data again based on the block ID.
- Upload status
 - Request mode: synchronous.
 - Call the `getStatus` method to obtain the current upload status.
 - Call the `getBlockList` method to obtain the blocks that are uploaded. Compare the result with the list of block IDs that were previously sent to the server and re-upload the blocks that fail to be uploaded.
- Upload termination
 - Request mode: synchronous.
 - Call the Commit(Long[] blocks) method. The blocks parameter indicates the blocks that are uploaded. The server verifies the block list.
 - Verification enhances data accuracy. If the provided list of block IDs is different from the list on the server, an error is returned.

- If the commit operation fails, try again.
- State description
 - UNKNOWN: This is the initial state when the server creates a session.
 - NORMAL: The upload session is created.
 - CLOSING: When you call the complete method to end an upload session, the server changes the state to CLOSING.
 - CLOSED: The data upload is complete. The data is moved to the directory where the result table is saved.
 - EXPIRED: The upload session times out.
 - CRITICAL: An error occurs.

1.7.4.5. DownloadSession

This topic describes the DownloadSession API.

This API is used to download data from MaxCompute tables.

Definition

The following code defines the DownloadSession API:

```
public class DownloadSession {
    DownloadSession(Configuration conf, String projectName, String tableName,
        String partitionSpec) throws TunnelException
    DownloadSession(Configuration conf, String projectName, String tableName,
        String partitionSpec, String downloadId) throws TunnelException
    public String getId()
    public long getRecordCount()
    public TableSchema getSchema()
    public TableTunnel.DownloadStatus getStatus()
    public RecordReader openRecordReader(long start, long count)
    public RecordReader openRecordReader(long start, long count, boolean compress)
}
```

Description

- Lifecycle: indicates the period from the time a download instance is created to the time data is downloaded.
- Download instance: You can call the constructor method or use TableTunnel to create a download instance.
 - Request mode: synchronous.
 - The server creates a session for the download instance and generates a unique download ID to identify the download instance. You can call the `getId` method on the client to obtain the download ID.
 - This operation results in high overheads. The server creates indexes for data files. If a large number of data files exists, it takes a long time to create indexes for the data files.
 - The server returns the total number of records. You can start multiple download sessions at the same time to download data based on the total number of data records.
- Data download
 - Request mode: asynchronous.

- Call the `openRecordReader` API to generate a `RecordReader` instance. The `start` parameter identifies the start position of the data record in this download session. The value of this parameter starts from 0 and must be greater than or equal to 0. The `count` parameter identifies the number of data records that are downloaded in this session. The value of the `count` parameter must be greater than 0.
- Download status
 - Request mode: synchronous.
 - Call the `getStatus` method to obtain the download status.
- Status description
 - UNKNOWN: This is the initial state when the server creates a download session.
 - NORMAL: The download object is created.
 - CLOSED: The download is complete.
 - EXPIRED: The download session times out.

1.7.4.6. TunnelBufferedWriter

This topic describes the `TunnelBufferedWriter` API.

This API is used to upload data.

The upload process is complex due to limits on block management and connection timeout on the server. The Tunnel SDK provides an enhanced `RecordWriter` of `TunnelBufferWriter` to simplify the upload process.

Definition

The following code defines the `TunnelBufferedWriter` API:

```
public class TunnelBufferedWriter implements RecordWriter {
    public TunnelBufferedWriter(TableTunnel.UploadSession session, CompressOption option) throws IOException;
    public long getTotalBytes();
    public void setBufferSize(long bufferSize);
    public void setRetryStrategy(RetryStrategy strategy);
    public void write(Record r) throws IOException;
    public void close() throws IOException;
}
```

Description

- Lifecycle: indicates the period from the time `RecordWriter` is created to the time data is upload.
- `TunnelBufferedWriter` instance: You can call `openBufferedWriter` of `UploadSession` to create a `TunnelBufferedWriter` instance.
- Data upload: If you call `write`, data records are first written to the local cache. After the cache is full, multiple data records are submitted to the server at a time to avoid a connection timeout. If data upload fails, the system automatically retries the upload operation.
- Upload termination: You can call `close` and then `commit` of `UploadSession` to terminate the upload process.
- Buffer control: You can call `setBufferSize` to change the memory (in bytes) occupied by the buffer. We recommend that you set the memory size to a value greater than or equal to 64 MB. This prevents excessive small files from being generated on the server, which may affect the processing performance. The value ranges from 1 MB to 1000 MB. The default value is 64 MB.
- Retry policy settings: The following policies are provided: `EXPONENTIAL_BACKOFF`, `LINEAR_BACKOFF`, and `CONSTANT_BACKOFF`. The following code snippet sets the number of `write` retries to 6. To avoid unnecessary retries, you can perform each retry after an exponential interval, such as 4s, 8s, 16s, 32s, 64s, and 128s. By default, the interval starts from 4s.

```
RetryStrategy retry
= new RetryStrategy(6, 4, RetryStrategy.BackoffStrategy.EXPONENTIAL_BACKOFF)
writer = (TunnelBufferedWriter) uploadSession.openBufferedWriter();
writer.setRetryStrategy(retry);
```

 **Note** We recommend that you retain the preceding settings.

1.7.5. Tunnel SDK example

1.7.5.1. Example of simple uploads

This topic provides an example of simple uploads.

Example:

```
import java.io.IOException;
import java.util.Date;
import com.aliyun.odps.Column;
import com.aliyun.odps.Odps;
import com.aliyun.odps.PartitionSpec;
import com.aliyun.odps.TableSchema;
import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.RecordWriter;
import com.aliyun.odps.tunnel.TableTunnel;
import com.aliyun.odps.tunnel.TunnelException;
import com.aliyun.odps.tunnel.TableTunnel.UploadSession;
public class UploadSample {
    private static String accessId = "<your access id>";
    private static String accessKey = "<your access Key>";
    private static String tunnelUrl = "<your tunnel endpoint>";
    private static String odpsUrl = "<your odps endpoint>";
    private static String project = "<your project>";
    private static String table = "<your table name>";
    private static String partition = "<your partition spec>";
    public static void main(String args[]) {
        Account account = new AliyunAccount(accessId, accessKey);
        Odps odps = new Odps(account);
        odps.setEndpoint(odpsUrl);
        odps.setDefaultProject(project);
        try {
            TableTunnel tunnel = new TableTunnel(odps);
            tunnel.setEndpoint(tunnelUrl);
            PartitionSpec partitionSpec = new PartitionSpec(partition);
            UploadSession uploadSession = tunnel.createUploadSession(project,
                table, partitionSpec);
            System.out.println("Session Status is : "
                + uploadSession.getStatus().toString());
            TableSchema schema = uploadSession.getSchema();
            // After data is prepared, open a writer to start writing data to a block.
            // If you write a small amount of data to a block, a large number of small files are generated. This severely deteriorates computing performance. We strongly recommend that you write more than 64 MB of data to a block at a time. The amount of data that can be written to the same block must be less than 100 GB.
            // You can calculate the total amount of data that is written to a block by using the following formula: Total amount of data = Average amount of data in each write operation × Number of write operations. The total amount of d
```

Amount of data = Average amount of data in each write operation × Number of write operations. The total amount of data must be greater than 64 MB, but less than 100 GB.

```
RecordWriter recordWriter = uploadSession.openRecordWriter(0);
Record record = uploadSession.newRecord();
for (int i = 0; i < schema.getColumns().size(); i++) {
    Column column = schema.getColumn(i);
    switch (column.getType()) {
        case BIGINT:
            record.setBigint(i, 1L);
            break;
        case BOOLEAN:
            record.setBoolean(i, true);
            break;
        case DATETIME:
            record.setDatetime(i, new Date());
            break;
        case DOUBLE:
            record.setDouble(i, 0.0);
            break;
        case STRING:
            record.setString(i, "sample");
            break;
        default:
            throw new RuntimeException("Unknown column type: "
                + column.getType());
    }
}
for (int i = 0; i < 10; i++) {
    // Write data to the server. A network transmission process is triggered each time 8 KB of data is written.
    // If no network transmission is performed within 120s, the server closes the connection. In this case, the writer
    // becomes unavailable and you must rewrite data.
    recordWriter.write(record);
}
recordWriter.close();
uploadSession.commit(new Long[]{0L});
System.out.println("upload success!");
} catch (TunnelException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
```

1.7.5.2. Example of simple downloads

This topic provides an example of simple downloads.

Example:

```
import java.io.IOException; import java.util.Date;
import com.aliyun.odps.Column; import com.aliyun.odps.Odps;
import com.aliyun.odps.PartitionSpec; import com.aliyun.odps.TableSchema; import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount; import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.RecordReader; import com.aliyun.odps.tunnel.TableTunnel;
import com.aliyun.odps.tunnel.TableTunnel.DownloadSession; import com.aliyun.odps.tunnel.TunnelException;
```

```
public class DownloadSample {
    private static String accessId = "<your access id>"; private static String accessKey = "<your access Key>";
    private static String tunnelUrl = "<your tunnel endpoint>";
    private static String odpsUrl = "<your odps endpoint>";
    private static String project = "<your project>"; private static String table = "<your table name>";
    private static String partition = "<your partition spec>";
    public static void main(String args[]) {
        Account account = new AliyunAccount(accessId, accessKey); Odps odps = new Odps(account); odps.setEndpoint(odpsUrl);
        odps.setDefaultProject(project);
        TableTunnel tunnel = new TableTunnel(odps); tunnel.setEndpoint(tunnelUrl);
        PartitionSpec partitionSpec = new PartitionSpec(partition); try {
            DownloadSession downloadSession = tunnel.createDownloadSession(project, table, partitionSpec);
            System.out.println("Session Status is : "
                + downloadSession.getStatus().toString());
            long count = downloadSession.getRecordCount(); System.out.println("RecordCount is: " + count);
            RecordReader recordReader = downloadSession.openRecordReader(0, count);
            Record record;
            while ((record = recordReader.read()) != null) { consumeRecord(record, downloadSession.getSchema());
            }
            recordReader.close();
        } catch (TunnelException e) { e.printStackTrace();
        } catch (IOException e1) { e1.printStackTrace();
        }
        }
        private static void consumeRecord(Record record, TableSchema schema) { for (int i = 0; i < schema.getColumns().size(); i++) {
            Column column = schema.getColumn(i); String colValue = null;
            switch (column.getType()) { case BIGINT: {
                Long v = record.getBigint(i);
                colValue = v == null ? null : v.toString(); break;
            }
            case BOOLEAN: {
                Boolean v = record.getBoolean(i); colValue = v == null ? null : v.toString(); break;
            }
            case DATETIME: {
                Date v = record.getDatetime(i); colValue = v == null ? null : v.toString(); break;
            }
            case DOUBLE: {
                Double v = record.getDouble(i); colValue = v == null ? null : v.toString(); break;
            }
            case STRING: {
                String v = record.getString(i);
                colValue = v == null ? null : v.toString(); break;
            }
            default:
                throw new RuntimeException("Unknown column type: "
                    + column.getType());
            }
            System.out.print(colValue == null ? "null" : colValue); if (i != schema.getColumns().size())
                System.out.print("\t");
            }
            System.out.println();
        }
    }
}
```

1.7.5.3. Example of multi-thread uploads

This topic provides an example of multi-thread uploads.

Example:

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.Date;
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import com.aliyun.odps.Column;
import com.aliyun.odps.Odps;
import com.aliyun.odps.PartitionSpec;
import com.aliyun.odps.TableSchema;
import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.RecordWriter;
import com.aliyun.odps.tunnel.TableTunnel;
import com.aliyun.odps.tunnel.TunnelException;
import com.aliyun.odps.tunnel.TableTunnel.UploadSession;
class UploadThread implements Callable<Boolean> {
    private long id;
    private RecordWriter recordWriter;
    private Record record;
    private TableSchema tableSchema;
    public UploadThread(long id, RecordWriter recordWriter, Record record, TableSchema tableSchema) {
        this.id = id;
        this.recordWriter = recordWriter;
        this.record = record;
        this.tableSchema = tableSchema;
    }
    @Override
    public Boolean call() {
        for (int i = 0; i < tableSchema.getColumns().size(); i++) {
            Column column = tableSchema.getColumn(i);
            switch (column.getType()) {
                case BIGINT:
                    record.setBigint(i, 1L);
                    break;
                case BOOLEAN:
                    record.setBoolean(i, true);
                    break;
                case DATETIME:
                    record.setDatetime(i, new Date());
                    break;
                case DOUBLE:
                    record.setDouble(i, 0.0);
                    break;
                case STRING:
                    record.setString(i, "sample");
                    break;
                default:
                    throw new RuntimeException("Unknown column type: "
                        + column.getType());
            }
        }
        return true;
    }
}
```

```
}
}
try {
    for (int i = 0; i < 10; i++) {
        // Write data to the server. A network transmission process is triggered each time 8 KB of data is written.
        // If no network transmission is performed within 120s, the server closes the connection. In this case, the writer
        // becomes unavailable and you must rewrite data.
        recordWriter.write(record);
    }
    recordWriter.close();
} catch (IOException e) {
    e.printStackTrace();
    return false;
}
return true;
}
}
public class UploadThreadSample {
    private static String accessId = "<your access id>";
    private static String accessKey = "<your access Key>";
    private static String tunnelUrl = "<your tunnel endpoint>";
    private static String odpsUrl = "<your odps endpoint>";
    private static String project = "<your project>";
    private static String table = "<your table name>";
    private static String partition = "<your partition spec>";
    private static int threadNum = 10;
    public static void main(String args[]) {
        Account account = new AliyunAccount(accessId, accessKey);
        Odps odps = new Odps(account);
        odps.setEndpoint(odpsUrl);
        odps.setDefaultProject(project);
        try {
            TableTunnel tunnel = new TableTunnel(odps);
            tunnel.setEndpoint(tunnelUrl);
            PartitionSpec partitionSpec = new PartitionSpec(partition);
            UploadSession uploadSession = tunnel.createUploadSession(project,
                table, partitionSpec);
            System.out.println("Session Status is : "
                + uploadSession.getStatus().toString());
            ExecutorService pool = Executors.newFixedThreadPool(threadNum);
            ArrayList<Callable<Boolean>> callers = new ArrayList<Callable<Boolean>>();
            // After data is prepared, open a writer to start writing data in multi-thread mode to a block.
            // If you write a small amount of data to a block, a large number of small files are generated. This severely deteriorates computing performance. We strongly recommend that you write more than 64 MB of data to a block at a time. The amount of data that can be written to the same block must be less than 100 GB.
            // You can calculate the total amount of data that is written to a block by using the following formula: Total amount of data = Average amount of data in each write operation × Number of write operations. The total amount of data must be greater than 64 MB, but less than 100 GB.
            for (int i = 0; i < threadNum; i++) {
                RecordWriter recordWriter = uploadSession.openRecordWriter(i);
                Record record = uploadSession.newRecord();
                callers.add(new UploadThread(i, recordWriter, record, uploadSession.getSchema()));
            }
            pool.invokeAll(callers);
            pool.shutdown();
            Long[] blockList = new Long[threadNum];
            for (int i = 0; i < threadNum; i++)
                blockList[i] = Long.valueOf(i);
        }
    }
}
```

```

        uploadSession.commit(blockList);
        System.out.println("upload success!");
    } catch (TunnelException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
}
}

```

1.7.5.4. Example of multi-thread downloads

This topic provides an example of multi-thread downloads.

Example:

```

import java.io.IOException;
import java.util.ArrayList; import java.util.Date; import java.util.List;
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutionException; import java.util.concurrent.ExecutorService; import java.util.concurrent.Executors;
import java.util.concurrent.Future;
import com.aliyun.odps.Column; import com.aliyun.odps.Odps;
import com.aliyun.odps.PartitionSpec; import com.aliyun.odps.TableSchema; import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount; import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.RecordReader; import com.aliyun.odps.tunnel.TableTunnel;
import com.aliyun.odps.tunnel.TableTunnel.DownloadSession; import com.aliyun.odps.tunnel.TunnelException;
class DownloadThread implements Callable<Long> { private long id;
private RecordReader recordReader; private TableSchema tableSchema;
public DownloadThread(int id,
RecordReader recordReader, TableSchema tableSchema) { this.id = id;
this.recordReader = recordReader; this.tableSchema = tableSchema;
}
@Override
public Long call() {
Long recordNum = 0L; try {
Record record;
while ((record = recordReader.read()) != null) { recordNum++;
System.out.print("Thread " + id + "\t"); consumeRecord(record, tableSchema);
}
recordReader.close();
} catch (IOException e) { e.printStackTrace();
}
return recordNum;
}
private static void consumeRecord(Record record, TableSchema schema) { for (int i = 0; i < schema.getColumns().size(); i++) {
Column column = schema.getColumn(i); String colValue = null;
switch (column.getType()) { case BIGINT: {
Long v = record.getBigint(i);
colValue = v == null ? null : v.toString(); break;
}
case BOOLEAN: {
}
}
}
}
}

```

```
Boolean v = record.getBoolean(i); colValue = v == null ? null : v.toString(); break;
}
case DATETIME: {
Date v = record.getDatetime(i); colValue = v == null ? null : v.toString(); break;
}
case DOUBLE: {
Double v = record.getDouble(i); colValue = v == null ? null : v.toString(); break;
}
case STRING: {
String v = record.getString(i);
colValue = v == null ? null : v.toString(); break;
}
default:
throw new RuntimeException("Unknown column type: "
+ column.getType());
}
System.out.print(colValue == null ? "null" : colValue); if (i != schema.getColumns().size())
System.out.print("\t");
}
System.out.println();
}
}
public class DownloadThreadSample {
private static String accessId = "<your access id>"; private static String accessKey = "<your access Key>";
private static String tunnelUrl = "<your tunnel endpoint>";
private static String odpsUrl = "<your odps endpoint>";
private static String project = "<your project>"; private static String table = "<your table name>";
private static String partition = "<your partition spec>";
private static int threadNum = 10; public static void main(String args[]) {
Account account = new AliyunAccount(accessId, accessKey);
Odps odps = new Odps(account); odps.setEndpoint(odpsUrl); odps.setDefaultProject(project);
TableTunnel tunnel = new TableTunnel(odps); tunnel.setEndpoint(tunnelUrl);
PartitionSpec partitionSpec = new PartitionSpec(partition); DownloadSession downloadSession;
try {
downloadSession = tunnel.createDownloadSession(project, table, partitionSpec);
System.out.println("Session Status is : "
+ downloadSession.getStatus().toString());
long count = downloadSession.getRecordCount(); System.out.println("RecordCount is: " + count);
ExecutorService pool = Executors.newFixedThreadPool(threadNum); ArrayList<Callable<Long>> callers = new Array
List<Callable<Long>>();
long start = 0;
long step = count / threadNum;
for (int i = 0; i < threadNum - 1; i++) {
RecordReader recordReader = downloadSession.openRecordReader( step * i, step);
callers.add(new DownloadThread( i, recordReader, downloadSession.getSchema()));
}
RecordReader recordReader = downloadSession.openRecordReader(step * (threadNum - 1), count
+ ((threadNum - 1) * step));
callers.add(new DownloadThread( threadNum - 1, recordReader, downloadSession.getSchema()));
Long downloadNum = 0L;
List<Future<Long>> recordNum = pool.invokeAll(callers); for (Future<Long> num : recordNum)
downloadNum += num.get(); System.out.println("Record Count is: " + downloadNum); pool.shutdown();
} catch (TunnelException e) { e.printStackTrace();
} catch (IOException e) { e.printStackTrace();
} catch (InterruptedException e) { e.printStackTrace();
} catch (ExecutionException e) { e.printStackTrace();
}
}
}
```

}

1.7.5.5. Example of uploading data by using BufferedWriter

This topic provides an example on how to upload data by using BufferedWriter of the MaxCompute Tunnel SDK.

Example:

```
// Initialize the code of MaxCompute and MaxCompute Tunnel.
RecordWriter writer = null;
TableTunnel.UploadSession uploadSession = tunnel.createUploadSession(projectName, tableName);
try {
    int i = 0;
    // Construct a BufferedWriter for MaxCompute Tunnel SDK.
    writer = uploadSession.openBufferedWriter();
    Record product = uploadSession.newRecord();
    for (String item : items) {
        product.setString("name", item);
        product.setBigint("id", i);
        // Call the write() method of BufferedWriter to write data.
        writer.write(product);
        i += 1;
    }
} finally {
    if (writer != null) {
        // Close BufferedWriter of MaxCompute Tunnel SDK.
        writer.close();
    }
}
// Commit the upload session to end the data upload.
uploadSession.commit();
```

1.7.5.6. Example of uploading data by using BufferedWriter in multithreading mode

This topic provides an example on how to upload data by using BufferedWriter of Tunnel SDK in multithreading mode.

Example:

```
class UploadThread extends Thread {
    private UploadSession session;
    private static int RECORD_COUNT = 1200;
    public UploadThread(UploadSession session) {
        this.session = session;
    }
    @Override
    public void run() {
        RecordWriter writer = up.openBufferedWriter();
        Record r = up.newRecord();
        for (int i = 0; i < RECORD_COUNT; i++) {
            r.setBigint(0, i);
            writer.write(r);
        }
        writer.close();
    }
};

public class Example {
    public static void main(String args[]) {
        // Initialize the code of MaxCompute and MaxCompute Tunnel.
        TableTunnel.UploadSession uploadSession = tunnel.createUploadSession(projectName, tableName);
        UploadThread t1 = new UploadThread(up);
        UploadThread t2 = new UploadThread(up);
        t1.start();
        t2.start();
        t1.join();
        t2.join();
        uploadSession.commit();
    }
}
```

1.7.5.7. Examples of uploading and downloading data of complex data types

This topic provides examples of uploading and downloading data of complex data types by using the SDK of MaxCompute Tunnel.

Upload data of complex data types

Example:

```

RecordWriter recordWriter = uploadSession.openRecordWriter(0);
ArrayRecord record = (ArrayRecord) uploadSession.newRecord();
// Prepare data.
List arrayData = Arrays.asList(1, 2, 3);
Map<String, Long> mapData = new HashMap<String, Long>();
mapData.put("a", 1L);
mapData.put("c", 2L);
List<Object> structData = new ArrayList<Object>();
structData.add("Lily");
structData.add(18);
// Pass data into a record.
record.setArray(0, arrayData);
record.setMap(1, mapData);
record.setStruct(2, new SimpleStruct((StructTypeInfo) schema.getColumn(2).getTypeInfo(), structData));
// Write the record to RecordWriter.
recordWriter.write(record);

```

Download data of complex data types

Example:

```

RecordReader recordReader = downloadSession.openRecordReader(0, 1);
// Read a record.
ArrayRecord record1 = (ArrayRecord)recordReader.read();
// Obtain data of the ARRAY type.
List field0 = record1.getArray(0);
List<Long> longField0 = record1.getArray(Long.class, 0);
// Obtain data of the MAP type.
Map field1 = record1.getMap(1);
Map<String, Long> typedField1 = record1.getMap(String.class, Long.class, 1);
// Obtain data of the STRUCT type.
Struct field2 = record1.getStruct(2);

```

Upload and download data of complex data types

Example:

```

import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import com.aliyun.odps.Odps;
import com.aliyun.odps.PartitionSpec;
import com.aliyun.odps.TableSchema;
import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount;
import com.aliyun.odps.data.ArrayRecord;
import com.aliyun.odps.data.RecordReader;
import com.aliyun.odps.data.RecordWriter;
import com.aliyun.odps.data.SimpleStruct;
import com.aliyun.odps.data.Struct;
import com.aliyun.odps.tunnel.TableTunnel;
import com.aliyun.odps.tunnel.TableTunnel.UploadSession;
import com.aliyun.odps.tunnel.TableTunnel.DownloadSession;
import com.aliyun.odps.tunnel.TableTunnel.UploadException;

```

```
import com.aliyun.odps.tunnel.TunnelException;
import com.aliyun.odps.type.StructTypeInfo;
public class TunnelComplexTypeSample {
    private static String accessId = "<your access id>";
    private static String accessKey = "<your access key>";
    private static String odpsUrl = "<your odps endpoint>";
    private static String project = "<your project>";
    private static String table = "<your table name>";
    // Partition in a partitioned table, such as "pt='1',ds='2'".
    // If the table is not a partitioned table, you do not need to execute the following statement.
    private static String partition = "<your partition spec>";
    public static void main(String args[]) {
        Account account = new AliyunAccount(accessId, accessKey);
        Odps odps = new Odps(account);
        odps.setEndpoint(odpsUrl);
        odps.setDefaultProject(project);
        try {
            TableTunnel tunnel = new TableTunnel(odps);
            PartitionSpec partitionSpec = new PartitionSpec(partition);
            //----- Upload data -----
            // Create an upload session for the table.
            // The table schema is {"col0": ARRAY<BIGINT>, "col1": MAP<STRING, BIGINT>, "col2": STRUCT<name:STRING, age:
            BIGINT>}.
            UploadSession uploadSession = tunnel.createUploadSession(project, table, partitionSpec);
            // Obtain the table schema.
            TableSchema schema = uploadSession.getSchema();
            // Open RecordWriter.
            RecordWriter recordWriter = uploadSession.openRecordWriter(0);
            ArrayRecord record = (ArrayRecord) uploadSession.newRecord();
            // Prepare data.
            List arrayData = Arrays.asList(1, 2, 3);
            Map<String, Long> mapData = new HashMap<String, Long>();
            mapData.put("a", 1L);
            mapData.put("c", 2L);
            List<Object> structData = new ArrayList<Object>();
            structData.add("Lily");
            structData.add(18);
            // Pass data into a record.
            record.setArray(0, arrayData);
            record.setMap(1, mapData);
            record.setStruct(2, new SimpleStruct((StructTypeInfo) schema.getColumn(2).getTypeInfo(), structData));
            // Write the record to RecordWriter.
            recordWriter.write(record);
            // Close RecordWriter.
            recordWriter.close();
            // Commit the upload session to complete the upload.
            uploadSession.commit(new Long[]{0L});
            System.out.println("upload success!");
            //----- Download data -----
            // Create a download session for the table.
            // The table schema is {"col0": ARRAY<BIGINT>, "col1": MAP<STRING, BIGINT>, "col2": STRUCT<name:STRING, age:
            BIGINT>}.
            DownloadSession downloadSession = tunnel.createDownloadSession(project, table, partitionSpec);
            schema = downloadSession.getSchema();
            // Open RecordReader.
            RecordReader recordReader = downloadSession.openRecordReader(0, 1);
            // Use RecordReader to read a record.
            ArrayRecord record1 = (ArrayRecord)recordReader.read();
```

```

// Obtain data of the ARRAY type.
List field0 = record1.getArray(0);
List<Long> longField0 = record1.getArray(Long.class, 0);
// Obtain data of the MAP type.
Map field1 = record1.getMap(1);
Map<String, Long> typedField1 = record1.getMap(String.class, Long.class, 1);
// Obtain data of the STRUCT type.
Struct field2 = record1.getStruct(2);
System.out.println("download success!");
} catch (TunnelException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
}
}

```

1.7.6. MaxCompute Streaming Tunnel

This topic describes the MaxCompute Streaming Tunnel service.

Description

MaxCompute Streaming Tunnel provides a new set of API operations and backend services. It provides an easy way for using streaming services, such as stream processing, tunnel, and data synchronization, as upstream services to write data to MaxCompute. MaxCompute Streaming Tunnel is introduced to resolve various issues that occur when streaming services call the original Batch API operation to write data. The issues include a large number of directory and file fragments, and a large number of request failures caused by highly concurrent uploads.

MaxCompute Streaming Tunnel provides the following features:

- Streaming semantics APIs: new APIs that simplify the development of distributed services.
- Point-to-point upload: provides higher performance and stability when network access is allowed.
- Automatic partition creation: allows you to create partitions for multiple distributed services at the same time.
- Limited data deduplication: deduplicates the data that you want to write based on the specified time or data volume.

 **Note** The automatic partition creation and limited data deduplication features will be available in later versions.

MaxCompute Streaming Tunnel outperforms other streaming services in the following aspects:

- Row-store file format: allows you to append data to a file to resolve the file fragmentation issue.
- Asynchronous data merging: improves the data storage efficiency without interrupting services.
- Separation of data requests from metadata operations: separates data write links from metadata operations (DDL operations). This ensures that DDL operations do not become a bottleneck in high queries per second (QPS) scenarios.
- Fine-grained throttling and resource allocation.

Limits

- Partition locking: MaxCompute Streaming Tunnel is used based on MaxCompute metadata. When MaxCompute writes streaming data to a table or partition, the table or partition is locked. During this

period, data modification operations, such as INSERT INTO and INSERT OVERWRITE, are not allowed. After the data write operation is complete, the table or partition is unlocked, and all operations can be performed.

- Delayed detection of DDL operations: Data links are separated from DDL operations. As a result, MaxCompute Streaming Tunnel detects DDL operations 30 seconds to 60 seconds after these operations are complete. For example, a data write operation to a table or partition is successful even after the table or partition has been deleted by using the DROP TABLE or DROP PARTITION statement. To prevent this issue, make sure that you execute the DROP TABLE or DROP PARTITION statement only after the data write operation is complete.
- Increased volume of stored hot data: To support asynchronous merging, MaxCompute Streaming Tunnel saves two copies of data that is written in the last hour. One copy is the original data, and the other copy is the merged data. Therefore, the volume of stored data increases to some extent on the day data is written.

Example

You can use the Maven repository to manage and configure the version of the SDK that you want to use. Sample configurations in the Maven repository:

```
<dependency>
<groupId>com.aliyun.odps</groupId>
<artifactId>odps-sdk-core</artifactId>
<version>0.36.2</version>
</dependency>
```

Sample code:

```
public class UploadSample {
    private static String accessId = "<your access id>";
    private static String accessKey = "<your access key>";
    private static String odpsUrl = "http://service.odps.aliyun.com/api";
    private static String project = "<your project>";
    private static String table = "<your table name>";
    private static String partition = "<your partition spec>";
    public static void main(String args[]) {
        Account account = new AliyunAccount(accessId, accessKey);
        Odps odps = new Odps(account);
        odps.setEndpoint(odpsUrl);
        odps.setDefaultProject(project);
        try {
            TableTunnel tunnel = new TableTunnel(odps);
            PartitionSpec partitionSpec = new PartitionSpec(partition);
            StreamUploadSession uploadSession = tunnel.createStreamUploadSession(project,
                table, partitionSpec);
            System.out.println("Session Status is : "
                + uploadSession.getStatus().toString());
            TableSchema schema = uploadSession.getSchema();
            StreamRecordPack pack = uploadSession.newRecordPack();
            Record record = uploadSession.newRecord();
            for (int i = 0; i < schema.getColumns().size(); i++) {
                Column column = schema.getColumn(i);
                switch (column.getType()) {
                    case BIGINT:
                        record.setBigint(i, 1L);
                        break;
                    case BOOLEAN:
```

```

        record.setBoolean(i, true);
        break;
    case DATETIME:
        record.setDatetime(i, new Date());
        break;
    case DOUBLE:
        record.setDouble(i, 0.0);
        break;
    case STRING:
        record.setString(i, "sample");
        break;
    default:
        throw new RuntimeException("Unknown column type: "
            + column.getType());
    }
}
for (int i = 0; i < 10; i++) {
    pack.append(record);
}
pack.flush();
System.out.println("upload success!");
} catch (TunnelException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
}

```

1.7.7. Appendix

1.7.7.1. FAQ related to data upload and download by using MaxCompute Tunnel

This topic provides answers to some frequently asked questions about data upload and download by using MaxCompute Tunnel.

What is MaxCompute Tunnel?

MaxCompute Tunnel is a data tunnel that is used to upload data to or download data from MaxCompute. View data cannot be uploaded or downloaded by using MaxCompute Tunnel.

Are duplicate block IDs allowed in an upload session?

No, each block ID in an upload session must be unique. If you use a block ID to open a RecordWriter, write data, and then call the close and commit methods in an upload session, you cannot use this block ID to open another RecordWriter. Valid values of block IDs: [0, 20000).

What is the maximum size of a block?

The maximum size of a block is 100 GB. We recommend that data of greater than 64 MB be written to each block. Each block corresponds to a file. We recommend that you write at least 64 MB of data to each block.

Can a session be shared among processes or threads? What is the lifecycle of a session?

Each session has a 24-hour lifecycle on the server. A session can be used within the 24 hours after it is created, and can be shared among processes or threads. The block ID of each session must be unique. You can perform the following steps to upload data:

1. Create a session.
2. Estimate the amount of data.
3. Assign blocks to threads, for example, assign blocks with the IDs of 0 to 100 to Thread 1 and assign blocks with the IDs of 100 to 200 to Thread 2.
4. Prepare data.
5. Upload data.
6. Commit all blocks to which data is written.

What do I do if read or write operations time out or an I/O exception occurs?

If you upload data, a network action is triggered each time the RecordWriter writes data of 8 KB. If no network action is triggered within 120 seconds, the server closes the connection, and the RecordWriter stops writing data. You must start another RecordWriter to write data.

If you download data, the RecordReader works in the similar way as the RecordWriter. If no network I/O occurs for a long period of time, the server closes the connection. We recommend that you perform read operations without calling other system API operations.

Which SDKs are supported by MaxCompute Tunnel?

MaxCompute Tunnel supports SDK for Java and SDK for C++.

Does MaxCompute Tunnel allow multiple clients to upload the same table at the same time?

Yes, MaxCompute Tunnel allows multiple clients to upload the same table at the same time.

Is MaxCompute Tunnel suitable for batch upload or streaming upload?

MaxCompute Tunnel is suitable for batch upload. It is not suitable for streaming upload.

Do I need to create partitions before I upload data by using MaxCompute Tunnel?

Yes, MaxCompute Tunnel does not automatically create partitions.

What is the relationship between Dship and MaxCompute Tunnel?

Dship is a tool that uploads and downloads data by using MaxCompute Tunnel.

Is data uploaded by using MaxCompute Tunnel appended to an existing file or is the data in the file overwritten by the uploaded data?

The uploaded data is appended to the existing file.

What is the routing feature of MaxCompute Tunnel?

The routing feature allows the SDK of MaxCompute Tunnel to obtain the endpoint of MaxCompute Tunnel by setting the MaxCompute endpoint. Therefore, you can run the SDK of MaxCompute Tunnel after you set the MaxCompute endpoint.

When data is uploaded by using MaxCompute Tunnel, what is the appropriate size of data in each block?

The size of data in each block is determined based on a number of factors, such as the network conditions, real-time performance requirements, data usage, and small files in clusters. In most cases, if the amount of data is large and data is continuously uploaded, the data size can be 64 MB to 256 MB. If data is uploaded once a day, the data size can be about 1 GB.

The timeout error message appears when data is downloaded by using MaxCompute Tunnel. Why?

This issue is usually caused by an incorrect endpoint. You can check the endpoint configuration by using different methods, for example, use Telnet to check network connectivity.

The following error message appears when data is downloaded by using MaxCompute Tunnel. Why?

```
You have NO privilege 'odps:Select' on {acs:odps:*:projects/XXX/tables/XXX}. project 'XXX' is protected
```

The reason is that data protection is enabled for the project from which data is downloaded. The data download you performed aims to transfer data from the project to another project. You can perform data download only after the project owner authorizes you to perform this operation.

The following error message appears when data is uploaded by using MaxCompute Tunnel. Why?

```
ErrorCode=FlowExceeded, ErrorMessage=Your flow quota is exceeded. **
```

The maximum number of concurrent upload or download requests is exceeded. By default, the quota for concurrent requests allowed by MaxCompute Tunnel is 2,000. A request counts against the quota after the request is sent and complete. If you encounter similar errors, we recommend that you use one of the following solutions:

- Make the system work in sleep mode and try again after the system awakes.
- Increase the quota for the project. Before you perform this operation, contact the administrator to estimate the traffic pressure.
- Report the issue to the project owner to locate and control the requests that consume most resources.

1.7.7.2. Common tunnel error codes

This topic describes common tunnel error codes.

The following table describes the common tunnel error codes.

ErrorCode	Cause	Recommended solution
NoSuchPartition	The partition does not exist.	Tunnel commands cannot be used to create partitions. Create partitions before you upload or download data.
InvalidProjectTable	The project name or table name is invalid.	Rename the project or table to ensure that a valid project or table name is used.
NoSuchProject	The specified project does not exist.	Check whether the project name is valid.

ErrorCode	Cause	Recommended solution
NoSuchTable	The specified table does not exist.	Check whether the table name is valid.
StatusConflict	The session expires or has been committed.	Recreate a session.
MalformedDataStream	The data format is invalid.	If the network connection is closed, reconnect it. If the schema is inconsistent with the table schema, make them consistent.
InvalidPartitionSpec	The partition information is invalid.	Check partition information. An example of a valid partition is <code>pt='1',ct='2017'</code> .
InvalidRowRange	The number of rows exceeds the upper limit or is 0.	Check related parameters.
Unauthorized	The account information, such as the AccessKey ID or AccessKey secret, is invalid, or the time gap between the local device and the server is more than 15 minutes.	Configure a valid AccessKey ID or AccessKey secret or make sure that the maximum time gap between the local device and the server is 15 minutes.
DataStoreError	A storage error occurs.	Contact the administrator.
NoPermission	You are not authorized to perform this operation or you have configured an IP address whitelist.	Check whether the granted permissions are correct.
MissingPartitionSpec	The partition information is not specified. If you want to perform operations on a partitioned table, the partition information must be specified.	Specify the partition information.
TableModified	Table data is modified by other tasks during data uploading or downloading.	Recreate a session.
FlowExceeded	The parallelism exceeds the quota.	Check and control parallelism. Before you increase the parallelism, contact the project owner or administrator to evaluate the traffic pressure.
InvalidResourceSpec	The information of the project, table, or partition is inconsistent with that specified for the session.	Check related information and try again later.
MethodNotAllowed	The method to export views is not supported.	Use other methods.
InvalidColumnSpec	The column name is invalid.	Rename the column to ensure that a valid column name is specified for data downloading.
DataVersionConflict	Cross-cluster replication is performed.	Try again later.

ErrorCode	Cause	Recommended solution
InternalServerError	An internal error occurs.	Try again later or contact the administrator.

1.8. MaxCompute MapReduce

1.8.1. Overview

1.8.1.1. MapReduce

This topic describes the MapReduce programming interfaces supported by MaxCompute.

MaxCompute provides the following types of MapReduce programming interfaces:

- **MaxCompute MapReduce:** MaxCompute native programming interfaces. These interfaces run fast and are easy to develop a program without exposing file systems.
- **Extended MaxCompute MapReduce (MR2):** an extension of MaxCompute MapReduce. This type of interface supports the logic to schedule complex jobs. The implementation method of these interfaces is the same as that used by the programming interfaces of MaxCompute MapReduce.
- **Hadoop-compatible MapReduce:** the programming interfaces that are highly compatible with Hadoop MapReduce. This type of interface is not compatible with MR2.

Note

- The basic concepts, job submission, input and output, and resource usage of the three types of interfaces are similar. The only difference lies in SDKs for Java.
- You cannot use MapReduce to read data from or write data to external tables.

Scenarios

MapReduce supports the following scenarios:

- **Search:** web crawling, inverted index, and PageRank.
- **Analysis of web access logs:**
 - Analyze and summarize the characteristics of user behavior, such as web browsing and online shopping. The analysis can be used to deliver personalized recommendations.
 - Analyze user access behavior.
- **Statistical analysis of texts:**
 - Word count and term frequency-inverse document frequency (TFIDF) analysis of popular novels.
 - Statistical analysis of references to academic papers and patent documents.
 - Wikipedia data analysis.
- **Mining of large amounts of data:** mining of unstructured data, spatio-temporal data, and image data.
- **Machine learning:** supervised learning, unsupervised learning, and classification algorithms, such as decision trees and support vector machines (SVMs).
- **Natural language processing (NLP):**
 - Training and forecast based on big data.
 - Construction of a co-occurrence matrix, mining of frequent itemset data, and duplicate document detection based on existing libraries.
- **Advertisement recommendations:** forecast of the click-through rate (CTR) and conversion rate (CVR).

Procedure

A MapReduce program processes data in two stages in sequence: the map and reduce stages. You can specify the logic to process data in the map and reduce stages. However, the logic must comply with the conventions of the MapReduce framework. The following procedure shows how the MapReduce framework processes data:

1. Before you perform map operations, make sure that input data is partitioned. After partitioning, the input data is divided into equally sized blocks that are called partitions. Each partition is processed as the input of a single map worker.
2. After partitioning, multiple map workers work at the same time. Each map worker reads its respective partition data, computes the data, and exports the result to a reduce worker.

 **Note** To generate data, a map worker must specify a key for each output record. The key determines the reduce worker for which the data record is targeted. Multiple keys may correspond to a single reduce worker. Data records with the same key are sent to the same reduce worker. A single reduce worker may receive data records with different keys.

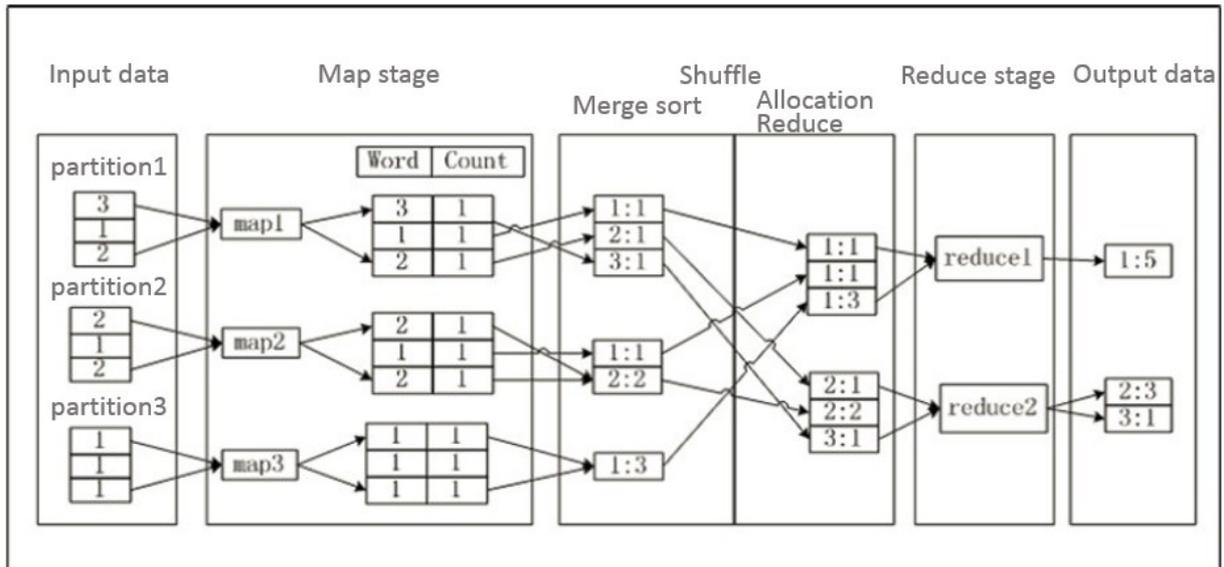
3. Before MapReduce enters the reduce stage, the MapReduce framework sorts data based on key values to make data records with the same key adjacent. If you specify a combiner, the MapReduce framework calls the combiner to combine data records with the same key.

 **Note** You can define the logic of the combiner. Unlike the typical MapReduce framework, MaxCompute requires the input and output parameters of the combiner to be consistent with reduce workers. This process is generally called shuffle.

4. When the MapReduce program enters the Reduce stage, data records with the same key are sent to the same reduce worker. A single reduce worker may receive data records from multiple map workers. Each reduce worker performs the reduce operation on multiple data records with the same key. After the reduce operation, all data records with the same key are converted into a single value.
5. The MapReduce framework generates the result.

The following example describes the concepts of MaxCompute MapReduce for WordCount at different stages.

Assume that a file named a.txt exists and each line of the file contains a digit. You want to count the number of times each digit appears. Each digit is called a word, and the number of times it is used represents the count. The following figure shows how MaxCompute MapReduce counts the words.



1. Partitions the a.txt file and uses data in each partition as the input of a single map worker.
2. For the map processing input, the Count parameter is set to 1 for each obtained word. The <Word, Count> pair is output as a word data key.
3. In the early phase of shuffle, the output of each map worker is sorted by key value (word value). After data records are sorted, the records are combined. To perform this operation, you must accumulate the count values that share the same key value to generate a <Word, Count> pair. This is a sorting and combining process.
4. In the late phase of shuffle, data is sent to the reduce workers. The reduce workers sort the received data records based on the key values.
5. In the reduce stage, each reduce worker uses the same logic as combiner to process data, and accumulates the count value with the same key value (word value) to obtain the output result.

Note All the MaxCompute data is stored in tables. Therefore, the input and output data of MaxCompute MapReduce can be saved only as tables. You cannot specify the output format, and no interfaces similar to file systems are provided.

1.8.1.2. MapReduce 2

This topic describes MapReduce 2 (MR2) supported by MaxCompute.

In MapReduce, data must be stored in a MaxCompute table or distributed file system, such as a Hadoop Distributed File System (HDFS), after each round of MapReduce operations. Typically, MapReduce runs multiple MapReduce jobs at the same time. After each job is complete, data is written to disks. However, data may need to be read only once in subsequent map operations to prepare for the shuffle operation. In this case, redundant I/O operations are performed on the disk.

The computing scheduling logic of MaxCompute supports more complicated programming models. Reduce operations can be consecutively performed without having a map operation in between. MaxCompute supports MR2 to allow reduce operations to be consecutively performed after a map operation.

MR2 supported by MaxCompute changes the underlying scheduling and I/O model to avoid redundant I/O operations when you run a job.

Hadoop ChainMapper and ChainReducer also support map or reduce operations in a chained fashion. However, they are essentially different from MR2.

Hadoop ChainMapper and ChainReducer follow the working mechanism used by MapReduce. They allow more than one map operation to be performed after a map or reduce operation. A map or reduce operation cannot be followed by reduce operations. This way, you can reuse the preceding map operation logic to split a map or reduce operation into multiple map stages. This does not change the underlying scheduling or I/O model.

1.8.1.3. Compatibility with Hadoop MapReduce

This topic describes the background information of compatibility with Hadoop MapReduce. It also describes how to use the Hadoop MapReduce plug-in.

MaxCompute provides a set of programming models and interfaces of Hadoop MapReduce. The input and output of the interfaces are data in MaxCompute tables. The data is organized as records, which demonstrate how the data is processed.

Programming interfaces of MaxCompute MapReduce differ from those of Hadoop MapReduce. To migrate Hadoop MapReduce jobs to MaxCompute MapReduce, you must rewrite the MapReduce code, compile and debug the code by calling MaxCompute MapReduce interfaces, package the final code into a JAR file, and then upload the file to MaxCompute. This process is tedious and labor-intensive for development and testing. It was expected that the original Hadoop MapReduce code can be used on MaxCompute with few modifications or without configurations.

To achieve the ideal solution, MaxCompute provides a plug-in to adapt Hadoop MapReduce to MaxCompute MapReduce. The plug-in enables Hadoop MapReduce jobs to be compatible with MaxCompute MapReduce at the binary level. You can directly run the original Hadoop MapReduce JAR files on MaxCompute after you configure relevant settings. Code rewriting is not required. The plug-in is under testing and does not support custom comparators or key types.

The following procedure describes how to use the Hadoop MapReduce plug-in with the WordCount program.

Note

- For more information about compatibility with Hadoop MapReduce, see [Compatibility with Hadoop MapReduce](#).
- For more information about the Hadoop MapReduce SDK, see the [MapReduce official documentation](#).

Download the Hadoop MapReduce plug-in

Download the [Hadoop MapReduce](#) plug-in file, `openmr_hadoop2openmr-1.0.jar`.

 **Note** This JAR file contains the dependencies of Hadoop 2.7.2. To avoid version conflicts, you must not include Hadoop dependencies in the JAR files of your jobs.

Prepare the JAR file

Compile and export the WordCount JAR file `wordcount_test.jar`. Sample source code of the WordCount program:

```

package com.aliyun.odps.mapred.example.hadoop;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;
import java.util.StringTokenizer;
public class WordCount {
    public static class TokenizerMapper
        extends Mapper<Object, Text, Text, IntWritable>{
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();
        public void map(Object key, Text value, Context context
        ) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }
    public static class IntSumReducer
        extends Reducer<Text,IntWritable,Text,IntWritable> {
        private IntWritable result = new IntWritable();
        public void reduce(Text key, Iterable<IntWritable> values,
            Context context
        ) throws IOException, InterruptedException {
            int sum = 0;
            for (IntWritable val : values) {
                sum += val.get();
            }
            result.set(sum);
            context.write(key, result);
        }
    }
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "word count");
        job.setJarByClass(WordCount.class);
        job.setMapperClass(TokenizerMapper.class);
        job.setCombinerClass(IntSumReducer.class);
        job.setReducerClass(IntSumReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}

```

Prepare test data

1. Execute the following statements to create the input table `wc_in` and the output table `wc_out`:

```
create table if not exists wc_in(line string);
create table if not exists wc_out(key string, cnt bigint);
```

2. Prepare the data that you want to import from the `data.txt` file into the `wc_in` table.

Sample data in the `data.txt` file:

```
hello maxcompute
hello mapreduce
```

3. Run the following Tunnel command on the MaxCompute client to import the preceding data from the `data.txt` file to the `wc_in` table:

```
tunnel upload data.txt wc_in;
```

Configure the mapping between HDFS directories and MaxCompute tables

Configure the mapping between HDFS directories and MaxCompute tables in the `wordcount-table-res.conf` file. Sample configurations in the `wordcount-table-res.conf` file:

```

{
  "file:/foo": {
    "resolver": {
      "resolver": "com.aliyun.odps.mapred.hadoop2openmr.resolver.TextFileResolver",
      "properties": {
        "text.resolver.columns.combine.enable": "true",
        "text.resolver.seperator": "\t"
      }
    },
    "tableInfos": [
      {
        "tblName": "wc_in",
        "partSpec": {},
        "label": "__default__"
      }
    ],
    "matchMode": "exact"
  },
  "file:/bar": {
    "resolver": {
      "resolver": "com.aliyun.odps.mapred.hadoop2openmr.resolver.BinaryFileResolver",
      "properties": {
        "binary.resolver.input.key.class": "org.apache.hadoop.io.Text",
        "binary.resolver.input.value.class": "org.apache.hadoop.io.LongWritable"
      }
    },
    "tableInfos": [
      {
        "tblName": "wc_out",
        "partSpec": {},
        "label": "__default__"
      }
    ],
    "matchMode": "fuzzy"
  }
}

```

The wordcount-table-res.conf file is a JSON file that describes the mapping between HDFS directories and MaxCompute tables. You must configure the input and output data. Each HDFS directory requires you to configure the following parameters: resolver, tableInfos, and matchMode.

Parameters:

- resolver: specifies how to process data in files. The following built-in resolvers can be used: com.aliyun.odps.mapred.hadoop2openmr.resolver.TextFileResolver and com.aliyun.odps.mapred.hadoop2openmr.resolver.BinaryFileResolver. After you specify the resolver parameter, you must configure the required properties for the resolver to parse data.
 - TextFileResolver: regards the input or output as plaintext if data is of the plaintext type. If you specify the resolver parameter for input, you must configure the text.resolver.columns.combine.enable and text.resolver.seperator properties. If text.resolver.columns.combine.enable is set to true, all columns in the input table are combined into a single string based on the delimiter specified by text.resolver.seperator. Otherwise, the first two columns in the input table are used to list keys and values.

- `BinaryFileResolver`: converts binary data into a data type that is supported by MaxCompute, such as `BIGINT`, `BOOLEAN`, and `DOUBLE`. If you specify the resolver parameter for output, you must configure the `binary.resolver.input.key.class` and `binary.resolver.input.value.class` properties. `binary.resolver.input.key.class` specifies the key class of the intermediate result, and `binary.resolver.input.value.class` indicates the value class of the intermediate result.
- `tableInfos`: the MaxCompute table that maps to an HDFS directory. Only the `tblName` parameter can be configured. You must configure the `partSpec` and `label` parameters the same as those in the preceding example.
- `matchMode`: specifies how MaxCompute tables map to HDFS directories. It can be set to `exact` or `fuzzy`. If this parameter is set to `fuzzy`, you can use a regular expression to map tables to HDFS directories.

Submit a job

Run the following command to submit a job on the MaxCompute client, `odpscmd`:

```
jar -DODPS_HADOOPMR_TABLE_RES_CONF=./wordcount-table-res.conf -classpath hadoop2openmr-1.0.jar,wordcount_test.jar com.aliyun.odps.mapred.example.hadoop.WordCount /foo/bar;
```

Note

- `wordcount-table-res.conf`: the configuration file that includes the `wc_in` and `wc_out` tables. The tables are mapped to the `/foo/bar` directory.
- `wordcount_test.jar`: the JAR file of the Hadoop MapReduce plug-in.
- `com.aliyun.odps.mapred.example.hadoop.WordCount`: the class name of the job that you want to run.
- `/foo/bar`: the HDFS directory, which is mapped to the `wc_in` and `wc_out` tables in the JSON configuration file.
- After you configure the mapping, you must use the Data Integration service of DataWorks to import the HDFS input file to the `wc_in` table for MapReduce computing, and export the `wc_out` table to the HDFS output directory `/bar`.
- To run the preceding command, make sure that you have stored the `hadoop2openmr-1.0.jar`, `wordcount_test.jar`, and `wordcount-table-res.conf` files in the `/foo/bar` directory of `odpscmd`. If you do not store the files in the `/foo/bar` directory, modify the configuration and the `-classpath` parameter in the command.

The following figure shows the running process of the job.

```

odps@ zhe> jar -DOPDS_TABLE_NAME=TEST_TBL1_CONF=wordcount-table-ms-conf -classpath hadoopcli.jar -Djava.library.path=/usr/lib64/jvm/java-1.8.0-212-linux-amd64/bin/wordcount.jar com.aliyun.odps.mapreduce.examples.wordcount.TestJob /usr/lib64/jvm/java-1.8.0-212-linux-amd64/bin/wordcount.jar
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/faq.html#staticLoggerBinder for further details.
Running job in console.
[INFO] deprecated - mapreduce.job.map.memory.mb is deprecated. Instead, use mapreduce.map.memory.mb
[INFO] deprecated - mapreduce.job.reduce.memory.mb is deprecated. Instead, use mapreduce.reduce.memory.mb
http://logview.odps.aliyun-inc.com:8080/login/?whwhhttp://28.381.214.148:8100/api?whwh=2816912883518705g1m6ak8tk6m=423u4821c1d1m4tt3k0m6u1504p9787118p5+PWF8T39K7jwkh11079k
1p6wsk311C2520w4k2j2310y30739633hcczmpgryh6p1Y30zL3pcc3p9mrm0j2044jw4j0r1x00310g1077m48118w9k11393w1w9yc21v41381311Pq=
InstanceID: 2816912883518705g1m6ak8tk6m=
[INFO] Job - the url to track the job: http://logview.odps.aliyun-inc.com:8080/login/?whwhhttp://28.381.214.148:8100/api?whwh=2816912883518705g1m6ak8tk6m=423u4821c1d1m4tt3k0m6u1504p9787118p5+PWF8T39K7jwkh11079k
401v41381311Pq=2816912883518705g1m6ak8tk6m=423u4821c1d1m4tt3k0m6u1504p9787118p5+PWF8T39K7jwkh11079k1Y30zL3pcc3p9mrm0j2044jw4j0r1x00310g1077m48118w9k11393w1w9yc21v41381311Pq=
...
2016-09-12 16:55:33 RL_3048-8/9/1[OK] RL_3_3048-8/9/1[OK]
2016-09-12 16:55:41 RL_3048-8/7/1[OK] RL_3_3048-8/9/1[OK]
...
Inputs:
  file_wc_in: 2 (408 bytes)
Outputs:
  file_wc_out: 3 (376 bytes)
RL_3048_2816912883518705g1m6ak8tk6m_107_8_8_8_3048:
  Number Counters:
  Input Records:
    Inputs: 2 (Cnt: 2, max: 2, avg: 2)
  Output Records:
    RL_3_1: 3 (Cnt: 3, max: 3, avg: 3)
RL_3_3048_2816912883518705g1m6ak8tk6m_107_8_8_8_3048:
  Number Counters:
  Input Records:
    Inputs: 3 (Cnt: 3, max: 3, avg: 3)
  Output Records:
    RL_3FS_OutsideOk: 3 (Cnt: 3, max: 3, avg: 3)
Counters: 0
OK
odps@ zhe>

```

After the job is complete, you can view the information in the wc_out table to check whether the job succeeds and whether the results meet your expectations.

```

odps@ zhe> read wc_out;
+-----+-----+
| key   | cnt |
+-----+-----+
| hello | 2   |
| mapreduce | 1 |
| maxcompute | 1 |
+-----+-----+
odps@ zhe>

```

1.8.2. Limits

This topic describes the limits of MapReduce.

- The memory consumed by a map or reduce worker can be 256 MB to 12 GB. By default, a single map or reduce worker occupies 2,048 MB memory of the MapReduce framework, and occupies 1,024 MB heap memory of a JVM.
- The number of resources referenced by a single task cannot exceed 256. A partitioned table is considered one resource. The number of input data records for a single task cannot exceed 1,024 and the number of output data records for a single task cannot exceed 256. The number of custom counters in a single task cannot exceed 64. The total bytes of resources referenced by a single task cannot exceed 2 GB.
- The number of map workers in a job is calculated by the MapReduce framework based on the split size. If no input table is specified, you can use odps.stage.mapper.num to specify the number of map workers. The number is in the range of 1 to 100,000.
- By default, the number of reduce workers in a job is one fourth of the number of map workers. You can set the number to a value that ranges from 0 to 2000. If reducers process much more data than mappers, the data processing in the reduce stage is time-consuming. A maximum of 2,000 reducers can be created.
- The maximum number of retries that are allowed for a map or reduce worker is 3. Exceptions that do not allow retries may cause jobs to fail.
- If the MapReduce program works in local mode, the default number of map workers is 2 and the default number of reduce workers is 1. The maximum number of map workers or reduce workers cannot exceed 100. The default number of download records is 100 and the number cannot exceed 10,000.
- The number of times that a map or reduce worker repeatedly reads a resource cannot exceed 64.

- The MapReduce framework determines the number of map workers based on the split size.
- Each column of the STRING type in a MaxCompute table cannot exceed 8 MB in length.
- If a map or reduce worker does not read or write data, or stops sending heartbeats by using `context.progress()`, the default timeout period is 600 seconds.
- When a MapReduce task references a table resource, the supported data types are BIGINT, DOUBLE, STRING, DATETIME, and BOOLEAN. An error is reported for tables of other data types.
- MapReduce does not read OSS data.
- MapReduce does not support the data types that are added to MaxCompute V2.0.

1.8.3. Features

1.8.3.1. Run command

This topic describes the command that is used to run MapReduce jobs.

The MaxCompute client provides a jar command for running MapReduce jobs.

Syntax:

```
Usage: jar [<GENERIC_OPTIONS>] <MAIN_CLASS> [ARGS]
-conf <configuration_file> Specify an application configuration file
-classpath <local_file_list> classpaths used to run mainClass
-D <name>=<value> Property value pair, which will be used to run mainClass
-local Run job in local mode
-resources <resource_name_list> file/table resources used in mapper or reducer, separate by comma
```

Parameters:

- `-conf <configuration file>`: the JobConf configuration file.
- `-classpath <local_file_list>`: the classpath that is used to run a MapReduce job in local mode. This parameter specifies the relative and absolute paths of the JAR package where the main function is located.
- `-D <prop_name>=<prop_value>`: the Java property of `<mainClass>` for a MaxCompute job that runs in local mode. You can specify multiple Java properties for a MaxCompute job.
- `-local`: specifies that a MapReduce job runs in local mode. This parameter is used for program debugging.
- `-resources <resource_name_list>`: the resources that are used to run a MapReduce job. You must specify the names of the resources in which the Map or Reduce function is located in `resource_name_list`.

 **Note** If the Map or Reduce function reads data from other MaxCompute resources, you must add the names of these resources to `resource_name_list`. Resource names in `resource_name_list` are separated by commas (,). If you use cross-project resources, add `PROJECT_NAME/resources/` before the resource name, for example, `resources otherproject/resources/resfile`.

You can use the `-conf <configuration file>` parameter to specify the JobConf file. This file contains the settings of JobConf in the SDK. Example of a JobConf file:

```
<configuration>
<property>
<name>import.filename</name>
<value>resource.txt</value>
</property>
</configuration>
```

In the preceding example, the JobConf file defines the `import.filename` variable. The value of this variable is `resource.txt`. You can obtain the value of this variable by calling the JobConf interface in MapReduce. You can also call the JobConf interface in the SDK for the same purpose.

Example:

```
jar -resources mapreduce-examples.jar -classpath mapreduce-examples.jar
org.alidata.odps.mr.examples.WordCount wc_in wc_out
add file data/src.txt
jar -resources src.txt,mapreduce-examples.jar -classpath mapreduce-examples.jar org.alidata.odps.mr.examples.W
ordCount wc_in wc_out
add file data/a.txt
add table wc_in as test_table add jar work.jar
jar -conf odps-mapred.xml -resources a.txt,test_table,work.jar
-classpath work.jar:otherlib.jar
-D import.filename=resource.txt org.alidata.odps.mr.examples.WordCount args
```

1.8.3.2. Concepts

1.8.3.2.1. Map/Reduce

This topic describes Map and Reduce, which are basic concepts in MaxCompute MapReduce.

If a map or reduce task runs, the `setup()`, `map()` or `reduce()`, and `cleanup()` methods are called. The `setup()` method is called before the `map()` or `reduce()` method. Each worker calls the `setup()` method only once. The `cleanup()` method is called after the `map()` or `reduce()` method. Each worker calls the `cleanup()` method only once.

 **Note** For more information about the usage examples, see [Sample programs](#).

1.8.3.2.2. Sorting

This topic describes sorting, a basic concept in MaxCompute MapReduce.

Some columns in the key records generated by a mapper can be used as sort columns. These columns do not support a custom comparator. You can select a few sort columns as group columns. These columns do not support a custom group comparator. Sort columns are used to sort your data, whereas group columns are used for secondary sorting.

 **Note** For detailed examples, see [Secondary sorting example](#).

1.8.3.2.3. Partition

This topic describes partition, a basic concept in MaxCompute MapReduce.

MaxCompute allows you to configure partition key columns and custom partitioners. Partition key columns take precedence over custom partitioners. Partitioners are used to allocate data generated by a mapper to different reducers based on the partition logic.

1.8.3.2.4. Combiner

This topic describes the combiner function, a basic concept in MaxCompute MapReduce.

The combiner function combines adjacent records at the shuffle stage. You can determine whether to use the combiner function based on your business logic.

The combiner function is the optimization of the MapReduce computing framework. The combiner logic is the same as the reducer logic. After a mapper generates data, the framework combines the data with the same key at the map stage.

 **Note** For detailed examples, see [Sample programs](#).

1.8.3.3. Submit a job

This topic describes how to run a jar command on the MaxCompute client to submit a MapReduce job.

The MaxCompute client provides a jar command to submit MapReduce jobs. Command syntax:

```
jar [<GENERIC_OPTIONS>] <MAIN_CLASS> [ARGS];
  -conf <configuration_file>   Specify an application configuration file
  -resources <resource_name_list> file\table resources used in mapper or reducer, separate by comma
  -classpath <local_file_list> classpaths used to run mainClass
  -D <name>=<value>           Property value pair, which will be used to run mainClass
  -l                           Run job in local mode
```

Parameters:

- `-conf <configuration file>`: the JobConf file. This file can affect the settings of JobConf in an SDK.

Template of a JobConf file:

```
<configuration>
  <property>
    <name>import.filename</name>
    <value>resource.txt</value>
  </property>
</configuration>
```

In the preceding template, the JobConf file defines the `import.filename` variable. The value of this variable is `resource.txt`. You can obtain the value of this variable over the JobConf interface in MapReduce. You can also call the JobConf interface in an SDK to obtain the value of the variable. For more information, see [Resource usage example](#).

Example:

```
add jar data\mapreduce-examples.jar;
jar -resources mapreduce-examples.jar -classpath data\mapreduce-examples.jar
  org.alidata.odps.mr.examples.WordCount wc_in wc_out;
add file data\src.txt;
add jar data\mapreduce-examples.jar;
jar -resources src.txt,mapreduce-examples.jar -classpath data\mapreduce-examples.jar
  org.alidata.odps.mr.examples.WordCount wc_in wc_out;
add file data\a.txt;
add table wc_in as test_table;
add jar data\work.jar;
jar -conf odps-mapred.xml -resources a.txt,test_table,work.jar
  -classpath data\work.jar:otherlib.jar
  -D import.filename=resource.txt org.alidata.odps.mr.examples.WordCount args;
```

- `-resources <resource_name_list>`: the resources that are used to run a MapReduce job. Typically, `resource_name_list` specifies the name of the resource in which the Map or Reduce function is located.

Note If the Map or Reduce function reads data from other MaxCompute resources, you must add the names of these resources to resource_name_list. The resource names are separated by commas (.). If you use cross-project resources, add PROJECT/resources/ before resource_name_list, for example, -resources otherproject/resources/resfile .

- `-classpath <local_file_list>`: the classpath that is used to run a MapReduce job in local mode. This parameter specifies the relative and absolute paths of the JAR packages that encapsulates the main function. Package names are separated by default file delimiters. In most cases, the Windows operating system uses semicolons (;) as the default file delimiter, and the Linux operating system uses commas (,) as the default file delimiter. If you run a MapReduce job on a cloud server, separate package names with commas (,).

Note The main function and Map/Reduce function are usually encapsulated into the same package. If you run the related program, `mapreduce-examples.jar` is specified in the following parameters: `-resources <resource_name_list>` and `-classpath <local_file_list>`. However, `-resources <resource_name_list>` references the Map or Reduce function and is run in a distributed environment, whereas `-classpath <local_file_list>` references the main function and is run in local mode with the specified JAR package saved in a local directory.

- `-D <name>=<value>`: the Java attribute of `<mainClass>` when you run a MapReduce job in local mode. You can define multiple Java attributes.
- `-l`: specifies that the MapReduce job is executed in local mode. This parameter is used for program debugging.

Example:

```
jar -conf \home\admin\myconf -resources a.txt,example.jar -classpath ..\lib\example.jar:.\other_lib.jar -D java.library.path=.\native;
```

1.8.3.4. Inputs and outputs

This topic describes the inputs and outputs of MapReduce jobs in MaxCompute.

- The inputs and outputs of MapReduce jobs in MaxCompute support built-in data types of MaxCompute, including BIGINT, DOUBLE, STRING, DATETIME, and BOOLEAN. User-defined data types are not supported.
- MapReduce supports input data from multiple tables with different schemas. You can use the map function to obtain the table information that corresponds to the current record.
- MapReduce supports null values as input data but does not support views as input data.
- A reduce job can write data to different tables or different partitions of a table. The destination tables or partitions can have different schemas. You can specify labels to distinguish outputs. If you want to use the default output, you do not need to specify a label. MapReduce does not support a function without output returned.

Note For more information about input and output examples, see [Example: Input and output data to multiple objects](#).

1.8.3.5. Use resources

This topic describes how MapReduce uses resources.

You can use the Map or Reduce function to read data from MaxCompute resources. A Map or Reduce worker loads resources to the memory for you to write code.

 **Note** For a detailed example, see [Resource usage example](#).

1.8.3.6. Job running in local mode

This topic describes the differences between the local mode and distributed mode in which MapReduce jobs run. It also provides examples of MapReduce jobs in local mode.

Introduction to the local mode

Before you run a job in local mode, you can specify the `-local` option in the JAR command to simulate the running of the job. This way, you can perform local debugging on the job.

During job running, the client downloads the metadata and data of the input table, metadata of the output table, and resources that are required for local debugging from MaxCompute. The downloaded data is saved to a local directory named *warehouse*.

After the job is completed, the computing results are saved to a file in the *warehouse* directory. If the input table and required resources are downloaded to the *warehouse* directory, MapReduce directly references the data and files in the directory next time, instead of downloading the data again.

Differences between the local mode and distributed mode

A MapReduce job that runs in local mode starts multiple map and reduce tasks to process data. These tasks run in sequence rather than in parallel. The simulated running process is different from an actual distributed running process in the following aspects:

- Rows in the input table: A maximum of 100 rows of data can be downloaded in local mode.
- Resource usage: In distributed mode, MaxCompute limits the size of resources that can be referenced. However, no limits are imposed on the size of resources in local mode.
- Security: MaxCompute MapReduce and user-defined functions (UDFs) are limited by a Java sandbox in distributed mode. However, no limits are imposed in local mode.

Examples

The following code shows an example of a MapReduce job in local mode:

```
odps:my_project> jar -l com.aliyun.odps.mapred.example.WordCount wc_in wc_out
Summary:
counters: 10
  map-reduce framework
    combine_input_groups=2
    combine_output_records=2
    map_input_bytes=4
    map_input_records=1
    map_output_records=2
    map_output_[wc_out]_bytes=0
    map_output_[wc_out]_records=0
    reduce_input_groups=2
    reduce_output_[wc_out]_bytes=8
    reduce_output_[wc_out]_records=2
OK
```

 **Note** For more information about the sample code of WordCount, see [WordCount example](#).

If this is the first time you run a local debugging command, a directory named *warehouse* is created in the current path after the command is executed. The following code shows the directory structure of *warehouse*.

```
<warehouse>
|__my_project (project directory)
|  |__<_tables_>
|  |  |__wc_in (table data directory)
|  |  |  |__data (file)
|  |  |  |
|  |  |  |__<_schema_> (file)
|  |  |  |
|  |  |  |__wc_out (table data directory)
|  |  |  |  |__data (file)
|  |  |  |  |
|  |  |  |  |__<_schema_> (file)
|  |  |  |
|  |  |__<_resources_>
|  |  |  |__table_resource_name (table resource)
|  |  |  |  |__<_ref_>
|  |  |  |
|  |  |  |__file_resource_name (file resource)
```

- Directories at the same level as *my_project* indicate projects. Directories at the same level as *wc_in* and *wc_out* indicate data tables. The table data that you read or write by using the JAR command is downloaded to directories at this level.
- The schema file stores the metadata of a table. The following code defines the file format:

```
project=local_project_name
table=local_table_name
columns=col1_name:col1_type,col2_name:col2_type
partitions=p1:STRING,p2:BIGINT -- In this example, you do not need to specify this field.
```

Note Separate the name and data type of a column with a colon (:). Separate columns with commas (.). The project and table names, *projectname.tablename*, must be declared at the beginning of the schema file. Separate the declaration and column definition with a comma (,), for example, *project_name.table_name,col1_name:col1_type,col2_name:col2_type,.....*

- The data file in the *tables* directory stores the table data. The number of columns and column data must match the definition in the schema file. Separate columns with commas (,).

For example, the schema file in the *wc_in* directory contains the following data:

```
my_project.wc_in,key:STRING,value:STRING
```

In this case, the data file contains the following data:

```
0,2
```

The client downloads the metadata and part of the data of a table from MaxCompute and saves the data to the preceding files. The next time you run this example program, the client directly uses the data in the *wc_in* directory, instead of downloading it again.

Note Data can be downloaded from MaxCompute only for MapReduce jobs that run in local mode.

For example, the schema file in the `wc_out` directory contains the following data:

```
my_project.wc_out,key:STRING,cnt:BIGINT
```

In this case, the data file contains the following data:

```
0,1  
2,1
```

The client downloads the metadata of the `wc_out` table from MaxCompute and saves the data to the schema file. After a job is completed, the results are saved to the data file.

 **Note**

- You can also edit the schema and data files and save the files in table directories.
- If you run a job in local mode and the client detects that the table directory exists, the client does not download the information of this table from MaxCompute. The local table directory can include a table that does not exist in MaxCompute.

1.8.4. SDK introduction

1.8.4.1. Overview of major MapReduce APIs

This topic describes major MapReduce APIs.

If you use Maven, you can obtain Maven dependencies from the [Maven Central Repository](#).

```
<dependency>  
  <groupId>com.aliyun.odps</groupId>  
  <artifactId>odps-sdk-mapred</artifactId>  
  <version>0.36.2-public</version>  
</dependency>
```

API	Description
MapperBase	The base class that custom Map functions must inherit. A mapper converts the records in the input table to key-value pairs and passes the key-value pairs to a reducer. Alternatively, a mapper can directly write the key-value pairs to a result table by skipping the Reduce stage. The jobs that skip the Reduce stage and directly return computing results are called map-only jobs.
ReducerBase	The base class that custom Reduce functions must inherit. A reducer reduces a set of values that are associated with a key.
TaskContext	Describes the context of a task. The task context is an input parameter of multiple member functions of MapperBase and ReducerBase.
JobClient	Submits and manages jobs. Jobs can be submitted in blocking mode (synchronous) or non-blocking mode (asynchronous).
RunningJob	Defines a running job. The objects of this class are used to track the instances on which MapReduce jobs are running.

API	Description
JobConf	Describes the configuration of a MapReduce job. The JobConf object is defined in the main function. Then, a job client submits a job to MaxCompute based on the JobConf object.

1.8.4.2. API description

1.8.4.2.1. MapperBase

This topic describes the main function interfaces of the MapperBase API.

The following table describes the main function interfaces of MapperBase.

Interface	Description
void cleanup(TaskContext context)	Calls the related method at the end of the map stage after the map method is called.
void map(long key, Record record, TaskContext context)	Calls the map method to process records in an input table.
void setup(TaskContext context)	Calls the related method at the beginning of the map stage before the map method is called.

1.8.4.2.2. ReducerBase

This topic describes the main function interfaces of the ReducerBase API.

The following table describes the main function interfaces of ReducerBase.

Interface	Description
void cleanup(TaskContext context)	Calls the related method at the end of the reduce stage after the reduce method is called.
void reduce(Record key, Iterator<Record > values, TaskContext context)	Calls the reduce method to process records in an input table.
void setup(TaskContext context)	Calls the related method at the beginning of the reduce stage before the reduce method is called.

1.8.4.2.3. TaskContext

This topic describes the main function interfaces of the TaskContext API.

The following table describes the main function interfaces of TaskContext.

Interface	Description
TableInfo[] getOutputTableInfo()	Gets output table information.
Record createOutputRecord()	Creates records for the default output table.
Record createOutputRecord(String label)	Creates records for the output table with the specified label.

Interface	Description
Record createMapOutputKeyRecord()	Creates records for keys in the key-value pairs that are generated at the map stage.
Record createMapOutputValueRecord()	Creates records for values in the key-value pairs that are generated at the map stage.
void write(Record record)	Writes records to the default output table. The interface can be called multiple times at the reduce stage.
void write(Record record, String label)	Writes records to the output table with the specified label. The interface can be called multiple times at the reduce stage.
void write(Record key, Record value)	Writes records to the intermediate results. The interface can be called multiple times at the map stage.
BufferedInputStream readResourceFileAsStream(String resourceName)	Reads a file resource.
Iterator<Record > readResourceTable(String resourceName)	Reads a table resource.
Counter getCounter(Enum<? > name)	Obtains the counter with the specified name.
Counter getCounter(String group, String name)	Obtains the counter with the specified name in the specified group.
void progress()	Sends heartbeat information to the MapReduce framework. If your task takes an extended period of time to process data and you do not need to call the framework during this time period, you can call this interface to avoid a task timeout. The default timeout period for a task is 600 seconds.

 **Note**

- The TaskContext API has a progress interface, which prevents it from being forced out due to timeout if a worker runs for a long time. This interface sends heartbeats to the framework rather than reporting the worker progress.
- The default timeout period for a worker is 10 minutes in MaxCompute MapReduce. You cannot change the timeout period. If a worker does not send heartbeat information by calling the progress interface in 10 minutes, the framework terminates the worker and the map or reduce task fails. Therefore, we recommend that you periodically call the progress interface in a map or reduce task to prevent the framework from terminating workers unexpectedly.

1.8.4.2.4. JobConf

This topic describes the main function interfaces of the JobConf API.

The following table describes the main function interfaces of JobConf.

Interface	Description
<code>void setResources(String resourceNames)</code>	Declares the resources that are used in the current job. A mapper or reducer can read only the resources that have been declared in the TaskContext object.
<code>void setMapOutputKeySchema(Column[] schema)</code>	Sets the attributes of keys that are passed from the mapper to the reducer.
<code>void setMapOutputValueSchema(Column[] schema)</code>	Sets the attributes of values that are passed from the mapper to the reducer.
<code>void setOutputKeySortColumns(String[] cols)</code>	Sets the columns for sorting the keys that are passed from the mapper to the reducer.
<code>void setOutputGroupingColumns(String[] cols)</code>	Sets the columns for grouping the keys.
<code>void setMapperClass(Class<? extends Mapper > > theClass)</code>	Sets a mapper for a job.
<code>void setPartitionColumns(String[] cols)</code>	Sets the partition key columns for a job. By default, the partition key columns are all columns of the keys that are generated by the mapper.
<code>void setReducerClass(Class<? extends Reducer > theClass)</code>	Sets a reducer for a job.
<code>void setCombinerClass(Class<? extends Reducer > theClass)</code>	Sets a combiner for a job. A combiner combines records with the same key. It is similar to a reducer but works at the map stage.
<code>void setSplitSize(long size)</code>	Sets the split size. Unit: MB. The default split size is 256 MB.
<code>void setNumReduceTasks(int n)</code>	Sets the number of reduce tasks. By default, the number of reduce tasks is one-fourth of the number of map tasks.
<code>void setMemoryForMapTask(int mem)</code>	Sets the memory available to a worker in a map task. Unit: MB. The default memory size is 2048 MB.
<code>void setMemoryForReduceTask(int mem)</code>	Sets the memory available to a worker in a reduce task. Unit: MB. The default memory size is 2048 MB.
<code>void setOutputSchema(Column[] schema, String label)</code>	Sets the output attribute of a designated label. If data is inserted into multiple objects, each output corresponds to a label.

 Note

- The grouping columns are selected from the sort columns. The sort columns and partition key columns must exist in keys.
- At the map stage, the hash values of records from a mapper are calculated based on the specified partition key columns. The hash values help determine the reducers to which records are passed. The records are sorted based on the sort columns before the records are passed to reducers.
- At the reduce stage, input records are grouped based on the grouping columns. Then, a group of records that share the same key are passed to the reduce method as input.

1.8.4.2.5. JobClient

This topic describes the main function interfaces of the JobClient API.

The following table describes the main function interfaces of JobClient.

Interface	Description
static RunningJob runJob(JobConf job)	Submits a MapReduce job in blocking (synchronous) mode and returns a RunningJob object.
static RunningJob submitJob(JobConf job)	Submits a MapReduce job in non-blocking (asynchronous) mode and returns a RunningJob object.

1.8.4.2.6. RunningJob

This topic describes the main function interfaces of the RunningJob API.

The following table describes the main function interfaces of RunningJob.

Interface	Description
String getInstanceID()	Obtains the ID of a job instance. You can use the job instance ID to view operational logs and manage jobs.
boolean isComplete()	Checks whether a job is completed.
boolean isSuccessful()	Checks whether a job instance is successful.
void waitForCompletion()	Waits for a job instance to end. The method is used for jobs that are submitted in asynchronous mode.
JobStatus getJobStatus()	Checks the running status of a job instance.
void killJob()	Ends the current job.
Counters getCounters()	Obtains the counter information.

1.8.4.2.7. InputUtils

This topic describes the main function interfaces of the InputUtils API.

The following table describes the main function interfaces of InputUtils.

Interface	Description
static void addTable(TableInfo table, JobConf conf)	Adds an input table to a task. The method can be called multiple times. New tables are appended to the input queue.
static void setTables(TableInfo [] tables, JobConf conf)	Adds multiple input tables to a task.

1.8.4.2.8. OutputUtils

This topic describes the main function interfaces of the OutputUtils API.

The following table describes the main function interfaces of OutputUtils.

Interface	Description
static void addTable(TableInfo table, JobConf conf)	Adds an output table to a task. The method can be called multiple times. New tables are appended to the output queue.
static void setTables(TableInfo [] tables, JobConf conf)	Adds multiple output tables to a task.

1.8.4.2.9. Pipeline

This topic describes the main interfaces of the Pipeline API.

Pipeline is the main class of MR2. You can call the Pipeline.builder method to build a pipeline. The following code shows the main interfaces of the Pipeline class:

```
public Builder addMapper(Class<? extends Mapper> mapper)
public Builder addMapper(Class<? extends Mapper> mapper,
    Column[] keySchema, Column[] valueSchema, String[] sortCols,
    SortOrder[] order, String[] partCols,
    Class<? extends Partitioner> theClass, String[] groupCols)
public Builder addReducer(Class<? extends Reducer> reducer)
public Builder addReducer(Class<? extends Reducer> reducer,
    Column[] keySchema, Column[] valueSchema, String[] sortCols,
    SortOrder[] order, String[] partCols,
    Class<? extends Partitioner> theClass, String[] groupCols)
public Builder setOutputKeySchema(Column[] keySchema)
public Builder setOutputValueSchema(Column[] valueSchema)
public Builder setOutputKeySortColumns(String[] sortCols)
public Builder setOutputKeySortOrder(SortOrder[] order)
public Builder setPartitionColumns(String[] partCols)
public Builder setPartitionerClass(Class<? extends Partitioner> theClass)
public Builder setOutputGroupingColumns(String[] cols)
```

The following example shows how to call the Pipeline.builder method to build a pipeline:

```

Job job = new Job();
Pipeline pipeline = Pipeline.builder()
    .addMapper(TokenizerMapper.class)
    .setOutputKeySchema(
        new Column[] { new Column("word", OdpsType.STRING) })
    .setOutputValueSchema(
        new Column[] { new Column("count", OdpsType.BIGINT) })
    .addReducer(SumReducer.class)
    .setOutputKeySchema(
        new Column[] { new Column("count", OdpsType.BIGINT) })
    .setOutputValueSchema(
        new Column[] { new Column("word", OdpsType.STRING),
            new Column("count", OdpsType.BIGINT) })
    .addReducer(IdentityReducer.class).createPipeline();
job.setPipeline(pipeline);
job.addInput(...)
job.addOutput(...)
job.submit();
    
```

As shown in the preceding example, you can create a MapReduce job in which a mapper is followed by two reducers in the Main function. If you are familiar with the basic features of MapReduce, MR2 is easy to use.



Note

- Before you use MR2, we recommend that you learn how to use MapReduce.
- You can create a MapReduce job in which a mapper is followed by only one reducer by using JobConf.

1.8.4.3. Compatibility with Hadoop MapReduce

This topic describes the compatibility between specific MaxCompute MapReduce interfaces and Hadoop MapReduce.

The following table describes whether specific MaxCompute MapReduce interfaces are compatible with Hadoop MapReduce.

Type	Interface	Compatible with Hadoop MapReduce
Mapper	void map(KEYIN key, VALUEIN value, org.apache.hadoop.mapreduce.Mapper.Context context)	Yes.
Mapper	void run(org.apache.hadoop.mapreduce.Mapper.Context context)	Yes.
Mapper	void setup(org.apache.hadoop.mapreduce.Mapper.Context context)	Yes.
Reducer	void cleanup(org.apache.hadoop.mapreduce.Reducer.Context context)	Yes.

Type	Interface	Compatible with Hadoop MapReduce
Reducer	void reduce(KEYIN key, VALUEIN value, org.apache.hadoop.mapreduce.Reducer.Context context)	Yes.
Reducer	void run(org.apache.hadoop.mapreduce.Reducer.Context context)	Yes.
Reducer	void setup(org.apache.hadoop.mapreduce.Reducer.Context context)	Yes.
Partitioner	int getPartition(KEY key, VALUE value, int numPartitions)	Yes.
MapContext, which extends TaskInputOutputContext	InputSplit getInputSplit()	No. An exception is reported.
ReduceContext	nextKey()	Yes.
ReduceContext	getValues()	Yes.
TaskInputOutputContext	getCurrentKey()	Yes.
TaskInputOutputContext	getCurrentValue()	Yes.
TaskInputOutputContext	getOutputCommitter()	No. An exception is reported.
TaskInputOutputContext	nextKeyValue()	Yes.
TaskInputOutputContext	write(KEYOUT key, VALUEOUT value)	Yes.
TaskAttemptContext	getCounter(Enum<?> counterName)	Yes.
TaskAttemptContext	getCounter(String groupName, String counterName)	Yes.
TaskAttemptContext	setStatus(String msg)	Empty implementation.
TaskAttemptContext	getStatus()	Empty implementation.
TaskAttemptContext	getTaskAttemptID()	No. An exception is reported.
TaskAttemptContext	getProgress()	No. An exception is reported.
TaskAttemptContext	progress()	Yes.
Job	addArchiveToClassPath(Path archive)	No.
Job	addCacheArchive(URI uri)	No.
Job	addCacheFile(URI uri)	No.

Type	Interface	Compatible with Hadoop MapReduce
Job	addFileToClassPath(Path file)	No.
Job	cleanupProgress()	No.
Job	createSymlink()	No. An exception is reported.
Job	failTask(TaskAttemptID taskId)	No.
Job	getCompletionPollInterval(Configuration conf)	Empty implementation.
Job	getCounters()	Yes.
Job	getFinishTime()	Yes.
Job	getHistoryUrl()	Yes.
Job	getInstance()	Yes.
Job	getInstance(Cluster ignored)	Yes.
Job	getInstance(Cluster ignored, Configuration conf)	Yes.
Job	getInstance(Configuration conf)	Yes.
Job	getInstance(Configuration conf, String jobName)	Empty implementation.
Job	getInstance(JobStatus status, Configuration conf)	No. An exception is reported.
Job	getJobFile()	No. An exception is reported.
Job	getJobName()	Empty implementation.
Job	getJobState()	No. An exception is reported.
Job	getPriority()	No. An exception is reported.
Job	getProgressPollInterval(Configuration conf)	Empty implementation.
Job	getReservationId()	No. An exception is reported.
Job	getSchedulingInfo()	No. An exception is reported.
Job	getStartTime()	Yes.
Job	getStatus()	No. An exception is reported.
Job	getTaskCompletionEvents(int startFrom)	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
Job	getTaskCompletionEvents(int startFrom, int numEvents)	No. An exception is reported.
Job	getTaskDiagnostics(TaskAttemptID taskId)	No. An exception is reported.
Job	getTaskOutputFilter(Configuration conf)	No. An exception is reported.
Job	getTaskReports(TaskType type)	No. An exception is reported.
Job	getTrackingURL()	Yes.
Job	isComplete()	Yes.
Job	isRetired()	No. An exception is reported.
Job	isSuccessful()	Yes.
Job	isUber()	Empty implementation.
Job	killJob()	Yes.
Job	killTask(TaskAttemptID taskId)	No.
Job	mapProgress()	Yes.
Job	monitorAndPrintJob()	Yes.
Job	reduceProgress()	Yes.
Job	setCacheArchives(URI[] archives)	No. An exception is reported.
Job	setCacheFiles(URI[] files)	No. An exception is reported.
Job	setCancelDelegationTokenUponJobCompletion(boolean value)	No. An exception is reported.
Job	setCombinerClass(Class<? extends Reducer> cls)	Yes.
Job	setCombinerKeyGroupingComparatorClass(Class<? extends RawComparator> cls)	Yes.
Job	setGroupingComparatorClass(Class<? extends RawComparator> cls)	Yes.
Job	setInputFormatClass(Class<? extends InputFormat> cls)	Empty implementation.
Job	setJar(String jar)	Yes.
Job	setJarByClass(Class<? > cls)	Yes.

Type	Interface	Compatible with Hadoop MapReduce
Job	setJobName(String name)	Empty implementation.
Job	setJobSetupCleanupNeeded(boolean needed)	Empty implementation.
Job	setMapOutputKeyClass(Class<?> theClass)	Yes.
Job	setMapOutputValueClass(Class<?> theClass)	Yes.
Job	setMapperClass(Class<? extends Mapper> cls)	Yes.
Job	setMapSpeculativeExecution(boolean speculativeExecution)	Empty implementation.
Job	setMaxMapAttempts(int n)	Empty implementation.
Job	setMaxReduceAttempts(int n)	Empty implementation.
Job	setNumReduceTasks(int tasks)	Yes.
Job	setOutputFormatClass(Class<? extends OutputFormat> cls)	No. An exception is reported.
Job	setOutputKeyClass(Class<?> theClass)	Yes.
Job	setOutputValueClass(Class<?> theClass)	Yes.
Job	setPartitionerClass(Class<? extends Partitioner> cls)	Yes.
Job	setPriority(JobPriority priority)	No. An exception is reported.
Job	setProfileEnabled(boolean newValue)	Empty implementation.
Job	setProfileParams(String value)	Empty implementation.
Job	setProfileTaskRange(boolean isMap, String newValue)	Empty implementation.
Job	setReducerClass(Class<? extends Reducer> cls)	Yes.
Job	setReduceSpeculativeExecution(boolean speculativeExecution)	Empty implementation.
Job	setReservationId(ReservationId reservationId)	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
Job	setSortComparatorClass(Class<? extends RawComparator> cls)	No. An exception is reported.
Job	setSpeculativeExecution(boolean speculativeExecution)	Yes.
Job	setTaskOutputFilter(Configuration conf, org.apache.hadoop.mapreduce.Job.TaskStatusFilter newValue)	No. An exception is reported.
Job	setUpProgress()	No. An exception is reported.
Job	setUser(String user)	Empty implementation.
Job	setWorkingDirectory(Path dir)	Empty implementation.
Job	submit()	Yes.
Job	toString()	No. An exception is reported.
Job	waitForCompletion(boolean verbose)	Yes.
Task Execution & Environment	mapreduce.map.java.opts	Empty implementation.
Task Execution & Environment	mapreduce.reduce.java.opts	Empty implementation.
Task Execution & Environment	mapreduce.map.memory.mb	Empty implementation.
Task Execution & Environment	mapreduce.reduce.memory.mb	Empty implementation.
Task Execution & Environment	mapreduce.task.io.sort.mb	Empty implementation.
Task Execution & Environment	mapreduce.map.sort.spill.percent	Empty implementation.
Task Execution & Environment	mapreduce.task.io.soft.factor	Empty implementation.
Task Execution & Environment	mapreduce.reduce.merge.inmem.thresholds	Empty implementation.
Task Execution & Environment	mapreduce.reduce.shuffle.merge.percent	Empty implementation.
Task Execution & Environment	mapreduce.reduce.shuffle.input.buffer.percent	Empty implementation.
Task Execution & Environment	mapreduce.reduce.input.buffer.percent	Empty implementation.
Task Execution & Environment	mapreduce.job.id	Empty implementation.
Task Execution & Environment	mapreduce.job.jar	Empty implementation.
Task Execution & Environment	mapreduce.job.local.dir	Empty implementation.

Type	Interface	Compatible with Hadoop MapReduce
Task Execution & Environment	mapreduce.task.id	Empty implementation.
Task Execution & Environment	mapreduce.task.attempt.id	Empty implementation.
Task Execution & Environment	mapreduce.task.is.map	Empty implementation.
Task Execution & Environment	mapreduce.task.partition	Empty implementation.
Task Execution & Environment	mapreduce.map.input.file	Empty implementation.
Task Execution & Environment	mapreduce.map.input.start	Empty implementation.
Task Execution & Environment	mapreduce.map.input.length	Empty implementation.
Task Execution & Environment	mapreduce.task.output.dir	Empty implementation.
JobClient	cancelDelegationToken(Token <DelegationTokenIdentifier> token)	No. An exception is reported.
JobClient	close()	Empty implementation.
JobClient	displayTasks(JobID jobId, String type, String state)	No. An exception is reported.
JobClient	getAllJobs()	No. An exception is reported.
JobClient	getCleanupTaskReports(JobID jobId)	No. An exception is reported.
JobClient	getClusterStatus()	No. An exception is reported.
JobClient	getClusterStatus(boolean detailed)	No. An exception is reported.
JobClient	getDefaultMaps()	No. An exception is reported.
JobClient	getDefaultReduces()	No. An exception is reported.
JobClient	getDelegationToken(Text renewer)	No. An exception is reported.
JobClient	getFs()	No. An exception is reported.
JobClient	getJob(JobID jobId)	No. An exception is reported.
JobClient	getJob(String jobId)	No. An exception is reported.
JobClient	getJobsFromQueue(String queueName)	No. An exception is reported.
JobClient	getMapTaskReports(JobID jobId)	No. An exception is reported.
JobClient	getMapTaskReports(String jobId)	No. An exception is reported.
JobClient	getQueueAclsForCurrentUser()	No. An exception is reported.
JobClient	getQueueInfo(String queueName)	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
JobClient	getQueues()	No. An exception is reported.
JobClient	getReduceTaskReports(JobID jobId)	No. An exception is reported.
JobClient	getReduceTaskReports(String jobId)	No. An exception is reported.
JobClient	getSetupTaskReports(JobID jobId)	No. An exception is reported.
JobClient	getStagingAreaDir()	No. An exception is reported.
JobClient	getSystemDir()	No. An exception is reported.
JobClient	getTaskOutputFilter()	No. An exception is reported.
JobClient	getTaskOutputFilter(JobConf job)	No. An exception is reported.
JobClient	init(JobConf conf)	No. An exception is reported.
JobClient	isJobDirValid(Path jobDirPath, FileSystem fs)	No. An exception is reported.
JobClient	jobsToComplete()	No. An exception is reported.
JobClient	monitorAndPrintJob(JobConf conf, RunningJob job)	No. An exception is reported.
JobClient	renewDelegationToken(Token<DelegationTokenIdentifier> token)	No. An exception is reported.
JobClient	run(String[] argv)	No. An exception is reported.
JobClient	runJob(JobConf job)	Yes.
JobClient	setTaskOutputFilter(JobClient.TaskStatusFilter newValue)	No. An exception is reported.
JobClient	setTaskOutputFilter(JobConf job, JobClient.TaskStatusFilter newValue)	No. An exception is reported.
JobClient	submitJob(JobConf job)	Yes.
JobClient	submitJob(String jobFile)	No. An exception is reported.
JobConf	deleteLocalFiles()	No. An exception is reported.
JobConf	deleteLocalFiles(String subdir)	No. An exception is reported.
JobConf	normalizeMemoryConfigValue(long val)	Empty implementation.
JobConf	setCombinerClass(Class<? extends Reducer> theClass)	Yes.

Type	Interface	Compatible with Hadoop MapReduce
JobConf	setCompressMapOutput(boolean compress)	Empty implementation.
JobConf	setInputFormat(Class<? extends InputFormat> theClass)	No. An exception is reported.
JobConf	setJar(String jar)	No. An exception is reported.
JobConf	setJarByClass(Class cls)	No. An exception is reported.
JobConf	setJobEndNotificationURI(String uri)	No. An exception is reported.
JobConf	setJobName(String name)	Empty implementation.
JobConf	setJobPriority(JobPriority prio)	No. An exception is reported.
JobConf	setKeepFailedTaskFiles(boolean keep)	No. An exception is reported.
JobConf	setKeepTaskFilesPattern(String pattern)	No. An exception is reported.
JobConf	setKeyFieldComparatorOptions(String keySpec)	No. An exception is reported.
JobConf	setKeyFieldPartitionerOptions(String keySpec)	No. An exception is reported.
JobConf	setMapDebugScript(String mDbgScript)	Empty implementation.
JobConf	setMapOutputCompressorClass(Class<? extends CompressionCodec> codecClass)	Empty implementation.
JobConf	setMapOutputKeyClass(Class<? > theClass)	Yes.
JobConf	setMapOutputValueClass(Class<? > theClass)	Yes.
JobConf	setMapperClass(Class<? extends Mapper> theClass)	Yes.
JobConf	setMapRunnerClass(Class<? extends MapRunnable> theClass)	No. An exception is reported.
JobConf	setMapSpeculativeExecution(boolean speculativeExecution)	Empty implementation.
JobConf	setMaxMapAttempts(int n)	Empty implementation.
JobConf	setMaxMapTaskFailuresPercent(int percent)	Empty implementation.

Type	Interface	Compatible with Hadoop MapReduce
JobConf	setMaxPhysicalMemoryForTask(long mem)	Empty implementation.
JobConf	setMaxReduceAttempts(int n)	Empty implementation.
JobConf	setMaxReduceTaskFailuresPercent(int percent)	Empty implementation.
JobConf	setMaxTaskFailuresPerTracker(int noFailures)	Empty implementation.
JobConf	setMaxVirtualMemoryForTask(long vmem)	Empty implementation.
JobConf	setMemoryForMapTask(long mem)	Yes.
JobConf	setMemoryForReduceTask(long mem)	Yes.
JobConf	setNumMapTasks(int n)	Yes.
JobConf	setNumReduceTasks(int n)	Yes.
JobConf	setNumTasksToExecutePerJvm(int numTasks)	Empty implementation.
JobConf	setOutputCommitter(Class<? extends OutputCommitter> theClass)	No. An exception is reported.
JobConf	setOutputFormat(Class<? extends OutputFormat> theClass)	Empty implementation.
JobConf	setOutputKeyClass(Class<? > theClass)	Yes.
JobConf	setOutputKeyComparatorClass(Class<? extends RawComparator> theClass)	No. An exception is reported.
JobConf	setOutputValueClass(Class<? > theClass)	Yes.
JobConf	setOutputValueGroupingComparator(Class<? extends RawComparator> theClass)	No. An exception is reported.
JobConf	setPartitionerClass(Class<? extends Partitioner> theClass)	Yes.
JobConf	setProfileEnabled(boolean newValue)	Empty implementation.
JobConf	setProfileParams(String value)	Empty implementation.

Type	Interface	Compatible with Hadoop MapReduce
JobConf	setProfileTaskRange(boolean isMap, String newValue)	Empty implementation.
JobConf	setQueueName(String queueName)	No. An exception is reported.
JobConf	setReduceDebugScript(String rDbgScript)	Empty implementation.
JobConf	setReducerClass(Class<? extends Reducer> theClass)	Yes.
JobConf	setReduceSpeculativeExecution(boolean speculativeExecution)	Empty implementation.
JobConf	setSessionId(String sessionId)	Empty implementation.
JobConf	setSpeculativeExecution(boolean speculativeExecution)	No. An exception is reported.
JobConf	setUseNewMapper(boolean flag)	Yes.
JobConf	setUseNewReducer(boolean flag)	Yes.
JobConf	setUser(String user)	Empty implementation.
JobConf	setWorkingDirectory(Path dir)	Empty implementation.
FileInputFormat	N/A	No. An exception is reported.
TextInputFormat	N/A	Yes.
InputSplit	mapred.min.split.size.	No. An exception is reported.
FileSplit	map.input.file	No. An exception is reported.
RecordWriter	N/A	No. An exception is reported.
RecordReader	N/A	No. An exception is reported.
OutputFormat	N/A	No. An exception is reported.
OutputCommitter	abortJob(JobContext jobContext, int status)	No. An exception is reported.
OutputCommitter	abortJob(JobContext context, JobStatus.State runState)	No. An exception is reported.
OutputCommitter	abortTask(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	abortTask(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	cleanupJob(JobContext jobContext)	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
OutputCommitter	cleanupJob(JobContext context)	No. An exception is reported.
OutputCommitter	commitJob(JobContext jobContext)	No. An exception is reported.
OutputCommitter	commitJob(JobContext context)	No. An exception is reported.
OutputCommitter	commitTask(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	needsTaskCommit(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	needsTaskCommit(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	setupJob(JobContext jobContext)	No. An exception is reported.
OutputCommitter	setupJob(JobContext jobContext)	No. An exception is reported.
OutputCommitter	setupTask(TaskAttemptContext taskContext)	No. An exception is reported.
OutputCommitter	setupTask(TaskAttemptContext taskContext)	No. An exception is reported.
Counter	getDisplayName()	Yes.
Counter	getName()	Yes.
Counter	getValue()	Yes.
Counter	increment(long incr)	Yes.
Counter	setValue(long value)	Yes.
Counter	setDisplayDisplayName(String displayName)	Yes.
DistributedCache	CACHE_ARCHIVES	No. An exception is reported.
DistributedCache	CACHE_ARCHIVES_SIZES	No. An exception is reported.
DistributedCache	CACHE_ARCHIVES_TIMESTAMPS	No. An exception is reported.
DistributedCache	CACHE_FILES	No. An exception is reported.
DistributedCache	CACHE_FILES_SIZES	No. An exception is reported.
DistributedCache	CACHE_FILES_TIMESTAMPS	No. An exception is reported.
DistributedCache	CACHE_LOCALARCHIVES	No. An exception is reported.
DistributedCache	CACHE_LOCALFILES	No. An exception is reported.
DistributedCache	CACHE_SYMLINK	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
DistributedCache	addArchiveToClassPath(Path archive, Configuration conf)	No. An exception is reported.
DistributedCache	addArchiveToClassPath(Path archive, Configuration conf, FileSystem fs)	No. An exception is reported.
DistributedCache	addCacheArchive(URI uri, Configuration conf)	No. An exception is reported.
DistributedCache	addCacheFile(URI uri, Configuration conf)	No. An exception is reported.
DistributedCache	addFileToClassPath(Path file, Configuration conf)	No. An exception is reported.
DistributedCache	addFileToClassPath(Path file, Configuration conf, FileSystem fs)	No. An exception is reported.
DistributedCache	addLocalArchives(Configuration conf, String str)	No. An exception is reported.
DistributedCache	addLocalFiles(Configuration conf, String str)	No. An exception is reported.
DistributedCache	checkURIs(URI[] uriFiles, URI[] uriArchives)	No. An exception is reported.
DistributedCache	createAllSymlink(Configuration conf, File jobCacheDir, File workDir)	No. An exception is reported.
DistributedCache	createSymlink(Configuration conf)	No. An exception is reported.
DistributedCache	getArchiveClassPaths(Configuration conf)	No. An exception is reported.
DistributedCache	getArchiveTimestamps(Configuration conf)	No. An exception is reported.
DistributedCache	getCacheArchives(Configuration conf)	No. An exception is reported.
DistributedCache	getCacheFiles(Configuration conf)	No. An exception is reported.
DistributedCache	getFileClassPaths(Configuration conf)	No. An exception is reported.
DistributedCache	getFileStatus(Configuration conf, URI cache)	No. An exception is reported.
DistributedCache	getFileTimestamps(Configuration conf)	No. An exception is reported.
DistributedCache	getLocalCacheArchives(Configuration conf)	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
DistributedCache	getLocalCacheFiles(Configuration conf)	No. An exception is reported.
DistributedCache	getSymlink(Configuration conf)	No. An exception is reported.
DistributedCache	getTimestamp(Configuration conf, URI cache)	No. An exception is reported.
DistributedCache	setArchiveTimestamps(Configuration conf, String timestamps)	No. An exception is reported.
DistributedCache	setCacheArchives(URI[] archives, Configuration conf)	No. An exception is reported.
DistributedCache	setCacheFiles(URI[] files, Configuration conf)	No. An exception is reported.
DistributedCache	setFileTimestamps(Configuration conf, String timestamps)	No. An exception is reported.
DistributedCache	setLocalArchives(Configuration conf, String str)	No. An exception is reported.
DistributedCache	setLocalFiles(Configuration conf, String str)	No. An exception is reported.
IsolationRunner	N/A	No. An exception is reported.
Profiling	N/A	Empty implementation.
Debugging	N/A	Empty implementation.
Data Compression	N/A	Yes.
Skipping Bad Records	N/A	No. An exception is reported.
Job Authorization	mapred.acls.enabled	No. An exception is reported.
Job Authorization	mapreduce.job.acl-view-job	No. An exception is reported.
Job Authorization	mapreduce.job.acl-modify-job	No. An exception is reported.
Job Authorization	mapreduce.cluster.administrators	No. An exception is reported.
Job Authorization	mapred.queue.queue-name.acl-administer-jobs	No. An exception is reported.
MultipleInputs	N/A	No. An exception is reported.
Multi{anchor:_GoBack}pleOutputs	N/A	Yes.
org.apache.hadoop.mapreduce.lib.db	N/A	No. An exception is reported.

Type	Interface	Compatible with Hadoop MapReduce
org.apache.hadoop.mapreduce.security	N/A	No. An exception is reported.
org.apache.hadoop.mapreduce.lib.jobcontrol	N/A	No. An exception is reported.
org.apache.hadoop.mapreduce.lib.chain	N/A	No. An exception is reported.
org.apache.hadoop.mapreduce.lib.db	N/A	No. An exception is reported.

1.8.5. Data types

This topic describes the data types supported by MapReduce.

MapReduce supports the following data types: BIGINT, DOUBLE, STRING, DATETIME, BOOLEAN, and DECIMAL. The following table lists the mappings between MaxCompute data types and Java data types.

MaxCompute SQL Type	Java Type
BIGINT	LONG
DOUBLE	DOUBLE
DECIMAL	BIGDECIMAL
BOOLEAN	BOOLEAN
STRING	STRING
DATETIME	DATE

1.8.6. Sample programs

1.8.6.1. WordCount example

This topic provides a WordCount example.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
```

```

public class WordCount {
    public static class TokenizerMapper extends MapperBase {
        private Record word;
        private Record one;
        @Override
        public void setup(TaskContext context) throws IOException {
            word = context.createMapOutputKeyRecord();
            one = context.createMapOutputValueRecord();
            one.set(new Object[] { 1L });
            System.out.println("TaskID:" + context.getTaskID().toString());
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            for (int i = 0; i < record.getColumnCount(); i++) {
                word.set(new Object[] { record.get(i).toString() });
                context.write(word, one);
            }
        }
    }
    /**
     * A combiner class that combines map output by sum them.
     */
    public static class SumCombiner extends ReducerBase {
        private Record count;
        @Override
        public void setup(TaskContext context) throws IOException {
            count = context.createMapOutputValueRecord();
        }
        // The combiner implements the same interface as that of the reducer. The combiner allows you to immediately
        // run a local reduce job on the mapper to reduce the output of the mapper.
        @Override
        public void reduce(Record key, Iterator<Record> values, TaskContext context)
            throws IOException {
            long c = 0;
            while (values.hasNext()) {
                Record val = values.next();
                c += (Long) val.get(0);
            }
            count.set(0, c);
            context.write(key, count);
        }
    }
    /**
     * A reducer class that just emits the sum of the input values.
     */
    public static class SumReducer extends ReducerBase {
        private Record result = null;
        @Override
        public void setup(TaskContext context) throws IOException {
            result = context.createOutputRecord();
        }
        @Override
        public void reduce(Record key, Iterator<Record> values, TaskContext context)
            throws IOException {
            long count = 0;
            while (values.hasNext()) {
                Record val = values.next();
            }
        }
    }
}

```

```
        record val = values.next(),
        count += (Long) val.get(0);
    }
    result.set(0, key.get(0));
    result.set(1, count);
    context.write(result);
}
}
public static void main(String[] args) throws Exception {
    if (args.length != 2) {
        System.err.println("Usage: WordCount <in_table> <out_table>");
        System.exit(2);
    }
    JobConf job = new JobConf();
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(SumCombiner.class);
    job.setReducerClass(SumReducer.class);
    // Configure the schema that defines the intermediate output of the mapper as key-value pairs. The intermediate output of the mapper exists as records.
    job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
    // Configure information about input and output tables.
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    JobClient.runJob(job);
}
}
```

1.8.6.2. MapOnly example

This topic provides a MapOnly example of MapReduce.

For MapOnly jobs, a mapper directly generates key-value pairs to MaxCompute tables. You need only to specify output tables. You do not need to specify the key-value metadata for the output of a mapper.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.SchemaUtils;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.data.TableInfo;
public class MapOnly {
    public static class MapperClass extends MapperBase {
        @Override
        public void setup(TaskContext context) throws IOException {
            boolean is = context.getJobConf().getBoolean("option.mapper.setup", false);
            // The main function executes the following logic only if option.mapper.setup is set to true in the JobConf file:
            if (is) {
                Record result = context.createOutputRecord();
                result.set(0, "setup");
                result.set(1, 1L);
                context.write(result);
            }
        }
    }
}
```

```

        context.write(result);
    }
}
@Override
public void map(long key, Record record, TaskContext context) throws IOException {
    boolean is = context.getJobConf().getBoolean("option.mapper.map", false);
    // The main function executes the following logic only if option.mapper.map is set to true in the JobConf file:
    if (is) {
        Record result = context.createOutputRecord();
        result.set(0, record.get(0));
        result.set(1, 1L);
        context.write(result);
    }
}
@Override
public void cleanup(TaskContext context) throws IOException {
    boolean is = context.getJobConf().getBoolean("option.mapper.cleanup", false);
    // The main function executes the following logic only if option.mapper.cleanup is set to true in the JobConf file:
    if (is) {
        Record result = context.createOutputRecord();
        result.set(0, "cleanup");
        result.set(1, 1L);
        context.write(result);
    }
}
}
public static void main(String[] args) throws Exception {
    if (args.length != 2 && args.length != 3) {
        System.err.println("Usage: OnlyMapper <in_table> <out_table> [setup|map|cleanup]");
        System.exit(2);
    }
    JobConf job = new JobConf();
    job.setMapperClass(MapperClass.class);
    // For MapOnly jobs, the number of reducers must be explicitly set to 0.
    job.setNumReduceTasks(0);
    // Configure information about input and output tables.
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    if (args.length == 3) {
        String options = new String(args[2]);
        // You can specify key-value pairs in the JobConf file, and use getJobConf of the context to query the configurations in a mapper.
        if (options.contains("setup")) {
            job.setBoolean("option.mapper.setup", true);
        }
        if (options.contains("map")) {
            job.setBoolean("option.mapper.map", true);
        }
        if (options.contains("cleanup")) {
            job.setBoolean("option.mapper.cleanup", true);
        }
    }
    JobClient.runJob(job);
}
}

```

1.8.6.3. MultipleInOut example

This topic provides a MultipleInOut example.

MaxCompute jobs can read data from multiple input tables and write data to multiple output tables. To read data from multiple input tables, you must make sure that the input tables have the same number of columns and the same data types. These requirements do not apply to output tables.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import java.util.LinkedHashMap;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 * Multi input & output example.
 **/
public class MultipleInOut {
    public static class TokenizerMapper extends MapperBase {
        Record word;
        Record one;
        @Override
        public void setup(TaskContext context) throws IOException {
            word = context.createMapOutputKeyRecord();
            one = context.createMapOutputValueRecord();
            one.set(new Object[] { 1L });
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            for (int i = 0; i < record.getColumnCount(); i++) {
                word.set(new Object[] { record.get(i).toString() });
                context.write(word, one);
            }
        }
    }
    public static class SumReducer extends ReducerBase {
        private Record result;
        private Record result1;
        private Record result2;
        @Override
        public void setup(TaskContext context) throws IOException {
            // Create a record for each output and add labels to distinguish outputs.
            result = context.createOutputRecord();
            result1 = context.createOutputRecord("out1");
            result2 = context.createOutputRecord("out2");
        }
        @Override
```

```

@Override
public void reduce(Record key, Iterator<Record> values, TaskContext context)
throws IOException {
    long count = 0;
    while (values.hasNext()) {
        Record val = values.next();
        count += (Long) val.get(0);
    }
    long mod = count % 3;
    if (mod == 0) {
        result.set(0, key.get(0));
        result.set(1, count);
        // If you do not specify a label, the default output is used.
        context.write(result);
    } else if (mod == 1) {
        result1.set(0, key.get(0));
        result1.set(1, count);
        context.write(result1, "out1");
    } else {
        result2.set(0, key.get(0));
        result2.set(1, count);
        context.write(result2, "out2");
    }
}

@Override
public void cleanup(TaskContext context) throws IOException {
    Record result = context.createOutputRecord();
    result.set(0, "default");
    result.set(1, 1L);
    context.write(result);
    Record result1 = context.createOutputRecord("out1");
    result1.set(0, "out1");
    result1.set(1, 1L);
    context.write(result1, "out1");
    Record result2 = context.createOutputRecord("out2");
    result2.set(0, "out2");
    result2.set(1, 1L);
    context.write(result2, "out2");
}

// Convert partition strings such as "ds=1/pt=2" to MAP.
public static LinkedHashMap<String, String> convertPartSpecToMap(
    String partSpec) {
    LinkedHashMap<String, String> map = new LinkedHashMap<String, String>();
    if (partSpec != null && !partSpec.trim().isEmpty()) {
        String[] parts = partSpec.split("/");
        for (String part : parts) {
            String[] ss = part.split("=");
            if (ss.length != 2) {
                throw new RuntimeException("ODPS-0730001: error part spec format: "
                    + partSpec);
            }
            map.put(ss[0], ss[1]);
        }
    }
    return map;
}

public static void main(String[] args) throws Exception {

```

```
String[] inputs = null;
String[] outputs = null;
if (args.length == 2) {
    inputs = args[0].split(",");
    outputs = args[1].split(",");
} else {
    System.err.println("MultipleInOut in... out...");
    System.exit(1);
}
JobConf job = new JobConf();
job.setMapperClass(TokenizerMapper.class);
job.setReducerClass(SumReducer.class);
job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
// Parse input table strings.
for (String in : inputs) {
    String[] ss = in.split("\\\\");
    if (ss.length == 1) {
        InputUtils.addTable(TableInfo.builder().tableName(ss[0]).build(), job);
    } else if (ss.length == 2) {
        LinkedHashMap<String, String> map = convertPartSpecToMap(ss[1]);
        InputUtils.addTable(TableInfo.builder().tableName(ss[0]).partSpec(map).build(), job);
    } else {
        System.err.println("Style of input: " + in + " is not right");
        System.exit(1);
    }
}
// Parse output table strings.
for (String out : outputs) {
    String[] ss = out.split("\\\\");
    if (ss.length == 1) {
        OutputUtils.addTable(TableInfo.builder().tableName(ss[0]).build(), job);
    } else if (ss.length == 2) {
        LinkedHashMap<String, String> map = convertPartSpecToMap(ss[1]);
        OutputUtils.addTable(TableInfo.builder().tableName(ss[0]).partSpec(map).build(), job);
    } else if (ss.length == 3) {
        if (ss[1].isEmpty()) {
            LinkedHashMap<String, String> map = convertPartSpecToMap(ss[2]);
            OutputUtils.addTable(TableInfo.builder().tableName(ss[0]).partSpec(map).build(), job);
        } else {
            LinkedHashMap<String, String> map = convertPartSpecToMap(ss[1]);
            OutputUtils.addTable(TableInfo.builder().tableName(ss[0]).partSpec(map)
                .label(ss[2]).build(), job);
        }
    } else {
        System.err.println("Style of output: " + out + " is not right");
        System.exit(1);
    }
}
JobClient.runJob(job);
}
```

1.8.6.4. Multijobs example

This topic provides a Multijobs example.

Example:

```

package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.RunningJob;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 * MultiJobs
 *
 * Running multiple job
 *
 **/
public class MultiJobs {
    public static class InitMapper extends MapperBase {
        @Override
        public void setup(TaskContext context) throws IOException {
            Record record = context.createOutputRecord();
            long v = context.getJobConf().getLong("multijobs.value", 2);
            record.set(0, v);
            context.write(record);
        }
    }
    public static class DecreaseMapper extends MapperBase {
        @Override
        public void cleanup(TaskContext context) throws IOException {
            // Obtain the variable values that are defined in the main function from JobConf.
            long expect = context.getJobConf().getLong("multijobs.expect.value", -1);
            long v = -1;
            int count = 0;
            // Read the data from the output table of the previous job.
            Iterator<Record> iter = context.readResourceTable("multijobs_res_table");
            while (iter.hasNext()) {
                Record r = iter.next();
                v = (Long) r.get(0);
                if (expect != v) {
                    throw new IOException("expect: " + expect + ", but: " + v);
                }
                count++;
            }
            if (count != 1) {
                throw new IOException("res_table should have 1 record, but: " + count);
            }
            Record record = context.createOutputRecord();
            v--;
            record.set(0, v);
            context.write(record);
            // Set the counter. The counter value can be obtained in the main function after the job is complete.
            context.getCounter("multijobs", "value").setValue(v);
        }
    }
}

```

```
    }
}
public static void main(String[] args) throws Exception {
    if (args.length != 1) {
        System.err.println("Usage: TestMultiJobs <table>");
        System.exit(1);
    }
    String tbl = args[0];
    long iterCount = 2;
    System.err.println("Start to run init job.");
    JobConf initJob = new JobConf();
    initJob.setLong("multijobs.value", iterCount);
    initJob.setMapperClass(InitMapper.class);
    InputUtils.addTable(TableInfo.builder().tableName("mr_empty").build(), initJob);
    OutputUtils.addTable(TableInfo.builder().tableName(tbl).build(), initJob);
    initJob.setMapOutputKeySchema(SchemaUtils.fromString("key:string"));
    initJob.setMapOutputValueSchema(SchemaUtils.fromString("value:string"));
    // Explicitly set the number of reducers to 0 for map-only jobs.
    initJob.setNumReduceTasks(0);
    JobClient.runJob(initJob);
    while (true) {
        System.err.println("Start to run iter job, count: " + iterCount);
        JobConf decJob = new JobConf();
        decJob.setLong("multijobs.expect.value", iterCount);
        decJob.setMapperClass(DecreaseMapper.class);
        InputUtils.addTable(TableInfo.builder().tableName("mr_empty").build(), decJob);
        OutputUtils.addTable(TableInfo.builder().tableName(tbl).build(), decJob);
        // Explicitly set the number of reducers to 0 for map-only jobs.
        decJob.setNumReduceTasks(0);
        RunningJob rJob = JobClient.runJob(decJob);
        iterCount--;
        // If the specified number of iterations is reached, exit the loop.
        if (rJob.getCounters().findCounter("multijobs", "value").getValue() == 0) {
            break;
        }
    }
    if (iterCount != 0) {
        throw new IOException("Job failed.");
    }
}
}
```

1.8.6.5. SecondarySort example

This topic provides a SecondarySort example.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
```

```

import com.aliyun.odps.mapred.utils.SchemaUtils;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.data.TableInfo;
/**
 *
 * This is an example ODPS Map/Reduce application. It reads the input table that
 * must contain two integers per record. The output is sorted by the first and
 * second number and grouped on the first number.
 *
 */
public class SecondarySort {
/**
 * Read two integers from each line and generate a key, value pair as ((left,
 * right), right).
 */
public static class MapClass extends MapperBase {
private Record key;
private Record value;
@Override
public void setup(TaskContext context) throws IOException {
key = context.createMapOutputKeyRecord();
value = context.createMapOutputValueRecord();
}
@Override
public void map(long recordNum, Record record, TaskContext context)
throws IOException {
long left = 0;
long right = 0;
if (record.getColumnCount() > 0) {
left = (Long) record.get(0);
if (record.getColumnCount() > 1) {
right = (Long) record.get(1);
}
}
key.set(new Object[] { (Long) left, (Long) right });
value.set(new Object[] { (Long) right });
context.write(key, value);
}
}
}
/**
 * A reducer class that just emits the sum of the input values.
 */
public static class ReduceClass extends ReducerBase {
private Record result = null;
@Override
public void setup(TaskContext context) throws IOException {
result = context.createOutputRecord();
}
@Override
public void reduce(Record key, Iterator<Record> values, TaskContext context)
throws IOException {
result.set(0, key.get(0));
while (values.hasNext()) {
Record value = values.next();
result.set(1, value.get(0));
context.write(result);
}
}
}

```

```
    }  
  }  
  public static void main(String[] args) throws Exception {  
    if (args.length != 2) {  
      System.err.println("Usage: secondarysrot <in> <out>");  
      System.exit(2);  
    }  
    JobConf job = new JobConf();  
    job.setMapperClass(MapClass.class);  
    job.setReducerClass(ReduceClass.class);  
    // Set multiple columns as keys.  
    //compare first and second parts of the pair  
    job.setOutputKeySortColumns(new String[] { "i1", "i2" });  
    //partition based on the first part of the pair  
    job.setPartitionColumns(new String[] { "i1" });  
    //grouping comparator based on the first part of the pair  
    job.setOutputGroupingColumns(new String[] { "i1" });  
    //the map output is LongPair, Long  
    job.setMapOutputKeySchema(SchemaUtils.fromString("i1:bigint,i2:bigint"));  
    job.setMapOutputValueSchema(SchemaUtils.fromString("i2x:bigint"));  
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);  
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);  
    JobClient.runJob(job);  
    System.exit(0);  
  }  
}
```

1.8.6.6. Resource usage example

This topic provides an example of using resources in MapReduce.

Example:

```
package com.aliyun.odps.mapred.open.example;  
import java.io.BufferedInputStream;  
import java.io.FileNotFoundException;  
import java.io.IOException;  
import com.aliyun.odps.data.Record;  
import com.aliyun.odps.data.TableInfo;  
import com.aliyun.odps.mapred.JobClient;  
import com.aliyun.odps.mapred.MapperBase;  
import com.aliyun.odps.mapred.TaskContext;  
import com.aliyun.odps.mapred.conf.JobConf;  
import com.aliyun.odps.mapred.utils.InputUtils;  
import com.aliyun.odps.mapred.utils.OutputUtils;  
import com.aliyun.odps.mapred.utils.SchemaUtils;  
/**  
 * Upload  
 *  
 * Import data from text file into table  
 *  
 **/  
public class Upload {  
  public static class UploadMapper extends MapperBase {  
    @Override  
    public void setup(TaskContext context) throws IOException {  
      Record record = context.createOutputRecord();
```

```

Record record = context.createOutputRecord();
StringBuilder importdata = new StringBuilder();
BufferedInputStream bufferedInput = null;
try {
    byte[] buffer = new byte[1024];
    int bytesRead = 0;
    String filename = context.getJobConf().get("import.filename");
    bufferedInput = context.readResourceFileAsStream(filename);
    while ((bytesRead = bufferedInput.read(buffer)) != -1) {
        String chunk = new String(buffer, 0, bytesRead);
        importdata.append(chunk);
    }
    String lines[] = importdata.toString().split("\n");
    for (int i = 0; i < lines.length; i++) {
        String[] ss = lines[i].split(",");
        record.set(0, Long.parseLong(ss[0].trim()));
        record.set(1, ss[1].trim());
        context.write(record);
    }
} catch (FileNotFoundException ex) {
    throw new IOException(ex);
} catch (IOException ex) {
    throw new IOException(ex);
} finally {
}
}
@Override
public void map(long recordNum, Record record, TaskContext context)
    throws IOException {
}
}
public static void main(String[] args) throws Exception {
    if (args.length != 2) {
        System.err.println("Usage: Upload <import_txt> <out_table>");
        System.exit(2);
    }
    JobConf job = new JobConf();
    job.setMapperClass(UploadMapper.class);
    // Specify the resource name, which can be obtained in the map stage by using the JobConf interface.
    job.set("import.filename", args[0]);
    // Explicitly set the number of reducers to 0 for MapOnly jobs.
    job.setNumReduceTasks(0);
    job.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("value:string"));
    InputUtils.addTable(TableInfo.builder().tableName("mr_empty").build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    JobClient.runJob(job);
}
}

```

Note

You can use one of the following methods to set the JobConf file:

- Use the JobConf interface in the SDK. This method is used in this example.
- Use the `-conf` parameter in a jar command to specify a new JobConf file.

1.8.6.7. Counter usage example

This topic provides an example of using counters in MapReduce.

Three counters are defined in this example: `map_outputs`, `reduce_outputs`, and `global_counts`. You can use the `setup`, `map`, `reduce`, and `cleanup` APIs of the `Map` or `Reduce` function to obtain custom counters and perform related operations.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.counter.Counter;
import com.aliyun.odps.counter.Counters;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.RunningJob;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.SchemaUtils;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.data.TableInfo;
/**
 *
 * User Defined Counters
 *
 */
public class UserDefinedCounters {
    enum MyCounter {
        TOTAL_TASKS, MAP_TASKS, REDUCE_TASKS
    }
    public static class TokenizerMapper extends MapperBase {
        private Record word;
        private Record one;
        @Override
        public void setup(TaskContext context) throws IOException {
            super.setup(context);
            Counter map_tasks = context.getCounter(MyCounter.MAP_TASKS);
            Counter total_tasks = context.getCounter(MyCounter.TOTAL_TASKS);
            map_tasks.increment(1);
            total_tasks.increment(1);
            word = context.createMapOutputKeyRecord();
            one = context.createMapOutputValueRecord();
            one.set(new Object[] { 1L });
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            for (int i = 0; i < record.getColumnCount(); i++) {
                word.set(new Object[] { record.get(i).toString() });
                context.write(word, one);
            }
        }
    }
}
public static class SumReducer extends ReducerBase {
```

```

public static class SumReducer extends ReducerBase {
    private Record result = null;
    @Override
    public void setup(TaskContext context) throws IOException {
        result = context.createOutputRecord();
        Counter reduce_tasks = context.getCounter(MyCounter.REDUCE_TASKS);
        Counter total_tasks = context.getCounter(MyCounter.TOTAL_TASKS);
        reduce_tasks.increment(1);
        total_tasks.increment(1);
    }
    @Override
    public void reduce(Record key, Iterator<Record> values, TaskContext context)
        throws IOException {
        long count = 0;
        while (values.hasNext()) {
            Record val = values.next();
            count += (Long) val.get(0);
        }
        result.set(0, key.get(0));
        result.set(1, count);
        context.write(result);
    }
}

public static void main(String[] args) throws Exception {
    if (args.length != 2) {
        System.err
            .println("Usage: TestUserDefinedCounters <in_table> <out_table>");
        System.exit(2);
    }
    JobConf job = new JobConf();
    job.setMapperClass(TokenizerMapper.class);
    job.setReducerClass(SumReducer.class);
    job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    RunningJob rJob = JobClient.runJob(job);
    // If the job is successful, the values of the custom counters in the job are returned.
    Counters counters = rJob.getCounters();
    long m = counters.findCounter(MyCounter.MAP_TASKS).getValue();
    long r = counters.findCounter(MyCounter.REDUCE_TASKS).getValue();
    long total = counters.findCounter(MyCounter.TOTAL_TASKS).getValue();
    System.exit(0);
}
}

```

1.8.6.8. Grep example

This topic provides a Grep example.

Example:

```

package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

```

```
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.Mapper;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.RunningJob;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 *
 * Extracts matching regexs from input files and counts them.
 *
 */
public class Grep {
/**
 * RegexMapper
 */
public class RegexMapper extends MapperBase {
private Pattern pattern;
private int group;
private Record word;
private Record one;
@Override
public void setup(TaskContext context) throws IOException {
JobConf job = (JobConf) context.getJobConf();
pattern = Pattern.compile(job.get("mapred.mapper.regex"));
group = job.getInt("mapred.mapper.regex.group", 0);
word = context.createMapOutputKeyRecord();
one = context.createMapOutputValueRecord();
one.set(new Object[] { 1L });
}
@Override
public void map(long recordNum, Record record, TaskContext context) throws IOException {
for (int i = 0; i < record.getColumnCount(); ++i) {
String text = record.get(i).toString();
Matcher matcher = pattern.matcher(text);
while (matcher.find()) {
word.set(new Object[] { matcher.group(group) });
context.write(word, one);
}
}
}
}
/**
 * LongSumReducer
 */
public class LongSumReducer extends ReducerBase {
private Record result = null;
@Override
public void setup(TaskContext context) throws IOException {
result = context.createOutputRecord();
}
@Override
public void reduce(Record key, Iterator<Record> values, TaskContext context) throws IOException {
```

```

        long count = 0;
        while (values.hasNext()) {
            Record val = values.next();
            count += (Long) val.get(0);
        }
        result.set(0, key.get(0));
        result.set(1, count);
        context.write(result);
    }
}
/**
 * A {@link Mapper} that swaps keys and values.
 **/
public class InverseMapper extends MapperBase {
    private Record word;
    private Record count;
    @Override
    public void setup(TaskContext context) throws IOException {
        word = context.createMapOutputValueRecord();
        count = context.createMapOutputKeyRecord();
    }
    /**
     * The inverse function. Input keys and values are swapped.
     **/
    @Override
    public void map(long recordNum, Record record, TaskContext context) throws IOException {
        word.set(new Object[] { record.get(0).toString() });
        count.set(new Object[] { (Long) record.get(1) });
        context.write(count, word);
    }
}
/**
 * IdentityReducer
 **/
public class IdentityReducer extends ReducerBase {
    private Record result = null;
    @Override
    public void setup(TaskContext context) throws IOException {
        result = context.createOutputRecord();
    }
    /** Writes all keys and values directly to output. **/
    @Override
    public void reduce(Record key, Iterator<Record> values, TaskContext context) throws IOException {
        result.set(0, key.get(0));
        while (values.hasNext()) {
            Record val = values.next();
            result.set(1, val.get(0));
            context.write(result);
        }
    }
}
public static void main(String[] args) throws Exception {
    if (args.length < 4) {
        System.err.println("Grep <inDir> <tmpDir> <outDir> <regex> [<group>"];
        System.exit(2);
    }
    JobConf grepJob = new JobConf();

```

```
grepJob.setMapperClass(RegexMapper.class);
grepJob.setReducerClass(LongSumReducer.class);
grepJob.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
grepJob.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), grepJob);
OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), grepJob);
// Set the regular expression for grepJob.
grepJob.set("mapred.mapper.regex", args[3]);
if (args.length == 5) {
    grepJob.set("mapred.mapper.regex.group", args[4]);
}
@SuppressWarnings("unused")
RunningJob rjGrep = JobClient.runJob(grepJob);
// Specify the output of grepJob as the input of sortJob.
JobConf sortJob = new JobConf();
sortJob.setMapperClass(InverseMapper.class);
sortJob.setReducerClass(IdentityReducer.class);
sortJob.setMapOutputKeySchema(SchemaUtils.fromString("count:bigint"));
sortJob.setMapOutputValueSchema(SchemaUtils.fromString("word:string"));
InputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), sortJob);
OutputUtils.addTable(TableInfo.builder().tableName(args[2]).build(), sortJob);
sortJob.setNumReduceTasks(1); // write a single file
sortJob.setOutputKeySortColumns(new String[] { "count" });
@SuppressWarnings("unused")
RunningJob rjSort = JobClient.runJob(sortJob);
}
}
```

1.8.6.9. Join example

This topic provides a Join example.

The MaxCompute MapReduce framework does not support Join operations. However, you can join data by using a custom map or reduce function.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 * Join, mr_Join_src1/mr_Join_src2(key bigint, value string), mr_Join_out(key
 * bigint, value1 string, value2 string)
 *
 *
 */
```

```

~/
public class Join {
    public static final Log LOG = LogFactory.getLog(Join.class);
    public static class JoinMapper extends MapperBase {
        private Record mapkey;
        private Record mapvalue;
        private long tag;
        @Override
        public void setup(TaskContext context) throws IOException {
            mapkey = context.createMapOutputKeyRecord();
            mapvalue = context.createMapOutputValueRecord();
            tag = context.getInputTableInfo().getLabel().equals("left") ? 0 : 1;
        }
        @Override
        public void map(long key, Record record, TaskContext context)
            throws IOException {
            mapkey.set(0, record.get(0));
            mapkey.set(1, tag);
            for (int i = 1; i < record.getColumnCount(); i++) {
                mapvalue.set(i - 1, record.get(i));
            }
            context.write(mapkey, mapvalue);
        }
    }
    public static class JoinReducer extends ReducerBase {
        private Record result = null;
        @Override
        public void setup(TaskContext context) throws IOException {
            result = context.createOutputRecord();
        }
        // Each input of the reduce function is the records that have the same key.
        @Override
        public void reduce(Record key, Iterator<Record> values, TaskContext context)
            throws IOException {
            long k = key.getBigint(0);
            List<Object[]> leftValues = new ArrayList<Object[]>();
            // Records are sorted based on the combination of the key and tag. This ensures that records in the left table are
            // passed to the reduce function first when the reduce function performs the Join operation.
            while (values.hasNext()) {
                Record value = values.next();
                long tag = (Long) key.get(1);
                // Data in the left table is first cached in memory.
                if (tag == 0) {
                    leftValues.add(value.toArray().clone());
                } else {
                    // Data in the right table is joined with all data in the left table.
                    // The sample code has poor performance and is only used as an example. We recommend that you do not
                    use the code in your production environment.
                    for (Object[] leftValue : leftValues) {
                        int index = 0;
                        result.set(index++, k);
                        for (int i = 0; i < leftValue.length; i++) {
                            result.set(index++, leftValue[i]);
                        }
                        for (int i = 0; i < value.getColumnCount(); i++) {
                            result.set(index++, value.get(i));
                        }
                    }
                    context.write(result);
                }
            }
        }
    }
}

```

```
    }
  }
}
}
}
public static void main(String[] args) throws Exception {
  if (args.length != 3) {
    System.err.println("Usage: Join <input table1> <input table2> <out>");
    System.exit(2);
  }
  JobConf job = new JobConf();
  job.setMapperClass(JoinMapper.class);
  job.setReducerClass(JoinReducer.class);
  job.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint,tag:bigint"));
  job.setMapOutputValueSchema(SchemaUtils.fromString("value:string"));
  job.setPartitionColumns(new String[]{"key"});
  job.setOutputKeySortColumns(new String[]{"key", "tag"});
  job.setOutputGroupingColumns(new String[]{"key"});
  job.setNumReduceTasks(1);
  InputUtils.addTable(TableInfo.builder().tableName(args[0]).label("left").build(), job);
  InputUtils.addTable(TableInfo.builder().tableName(args[1]).label("right").build(), job);
  OutputUtils.addTable(TableInfo.builder().tableName(args[2]).build(), job);
  JobClient.runJob(job);
}
}
```

1.8.6.10. Sleep example

This topic provides a Sleep example.

Example:

```

package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.conf.JobConf;
public class Sleep {
    private static final String SLEEP_SECS = "sleep.secs";
    public static class MapperClass extends MapperBase {
        // No data is entered, the map function is not executed, and the related logic can be written only into setup.
        @Override
        public void setup(TaskContext context) throws IOException {
            try {
                // Obtain the number of sleep seconds set in JobConf.
                Thread.sleep(context.getJobConf().getInt(SLEEP_SECS, 1) * 1000);
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
        }
    }
}
public static void main(String[] args) throws Exception {
    if (args.length != 1) {
        System.err.println("Usage: Sleep <sleep_secs>");
        System.exit(-1);
    }
    JobConf job = new JobConf();
    job.setMapperClass(MapperClass.class);
    // This instance is also a MapOnly job and the number of reducers must be set to 0.
    job.setNumReduceTasks(0);
    // The number of mappers must be specified by the user because no input table is provided.
    job.setNumMapTasks(1);
    job.set(SLEEP_SECS, args[0]);
    JobClient.runJob(job);
}
}

```

1.8.6.11. Unique example

This topic provides a Unique example.

Example:

```

package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 * Unique Remove duplicate words

```

```
*
**/
public class Unique {
    public static class OutputSchemaMapper extends MapperBase {
        private Record key;
        private Record value;
        @Override
        public void setup(TaskContext context) throws IOException {
            key = context.createMapOutputKeyRecord();
            value = context.createMapOutputValueRecord();
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            long left = 0;
            long right = 0;
            if (record.getColumnCount() > 0) {
                left = (Long) record.get(0);
                if (record.getColumnCount() > 1) {
                    right = (Long) record.get(1);
                }
            }
            key.set(new Object[] { (Long) left, (Long) right });
            value.set(new Object[] { (Long) left, (Long) right });
            context.write(key, value);
        }
    }
}

public static class OutputSchemaReducer extends ReducerBase {
    private Record result = null;
    @Override
    public void setup(TaskContext context) throws IOException {
        result = context.createOutputRecord();
    }
    @Override
    public void reduce(Record key, Iterator<Record> values, TaskContext context)
        throws IOException {
        result.set(0, key.get(0));
        while (values.hasNext()) {
            Record value = values.next();
            result.set(1, value.get(1));
        }
        context.write(result);
    }
}

public static void main(String[] args) throws Exception {
    if (args.length > 3 || args.length < 2) {
        System.err.println("Usage: unique <in> <out> [key|value|all]");
        System.exit(2);
    }
    String ops = "all";
    if (args.length == 3) {
        ops = args[2];
    }
    // The input group of Reduce is determined by the value of the setOutputGroupingColumns parameter. If this parameter is not specified, the default value MapOutputKeySchema is used.
    // Key Unique
    if (ops.equals("key")) {
        JobConf job = new JobConf();
    }
}
```

```

job.setMapperClass(OutputSchemaMapper.class);
job.setReducerClass(OutputSchemaReducer.class);
job.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint,value:bigint"));
job.setMapOutputValueSchema(SchemaUtils.fromString("key:bigint,value:bigint"));
job.setPartitionColumns(new String[] { "key" });
job.setOutputKeySortColumns(new String[] { "key", "value" });
job.setOutputGroupingColumns(new String[] { "key" });
job.set("tablename2", args[1]);
job.setNumReduceTasks(1);
job.setInt("table.counter", 0);
InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
JobClient.runJob(job);
}
// Key&Value Unique
if (ops.equals("all")) {
    JobConf job = new JobConf();
    job.setMapperClass(OutputSchemaMapper.class);
    job.setReducerClass(OutputSchemaReducer.class);
    job.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint,value:bigint"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("key:bigint,value:bigint"));
    job.setPartitionColumns(new String[] { "key" });
    job.setOutputKeySortColumns(new String[] { "key", "value" });
    job.setOutputGroupingColumns(new String[] { "key", "value" });
    job.set("tablename2", args[1]);
    job.setNumReduceTasks(1);
    job.setInt("table.counter", 0);
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    JobClient.runJob(job);
}
// Value Unique
if (ops.equals("value")) {
    JobConf job = new JobConf();
    job.setMapperClass(OutputSchemaMapper.class);
    job.setReducerClass(OutputSchemaReducer.class);
    job.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint,value:bigint"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("key:bigint,value:bigint"));
    job.setPartitionColumns(new String[] { "value" });
    job.setOutputKeySortColumns(new String[] { "value" });
    job.setOutputGroupingColumns(new String[] { "value" });
    job.set("tablename2", args[1]);
    job.setNumReduceTasks(1);
    job.setInt("table.counter", 0);
    InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), job);
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    JobClient.runJob(job);
}
}
}
}

```

1.8.6.12. Sort example

This topic provides a Sort example.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Date;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.TaskContext;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.example.lib.IdentityReducer;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
/**
 * This is the trivial map/reduce program that does absolutely nothing other
 * than use the framework to fragment and sort the input values.
 */
public class Sort {
    static int printUsage() {
        System.out.println("sort <input> <output>");
        return -1;
    }
    /**
     * Implements the identity function, mapping record's first two columns to
     * outputs.
     */
    public static class IdentityMapper extends MapperBase {
        private Record key;
        private Record value;
        @Override
        public void setup(TaskContext context) throws IOException {
            key = context.createMapOutputKeyRecord();
            value = context.createMapOutputValueRecord();
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            key.set(new Object[] { (Long) record.get(0) });
            value.set(new Object[] { (Long) record.get(1) });
            context.write(key, value);
        }
    }
    /**
     * The main driver for sort program. Invoke this method to submit the
     * map/reduce job.
     *
     * @throws IOException
     *     When there is communication problems with the job tracker.
     */
    public static void main(String[] args) throws Exception {
        JobConf jobConf = new JobConf();
        jobConf.setMapperClass(IdentityMapper.class);
        jobConf.setReducerClass(IdentityReducer.class);
        // For global sorting, the number of reducers is set to 1. All the data is transferred to the same reducer.
        // This method applies only to the scenarios when small amounts of data are processed. If large amounts of data need
        // to be processed, use other methods, such as TeraSort.
    }
}
```

```
jobConf.setNumReduceTasks(1);
jobConf.setMapOutputKeySchema(SchemaUtils.fromString("key:bigint"));
jobConf.setMapOutputValueSchema(SchemaUtils.fromString("value:bigint"));
InputUtils.addTable(TableInfo.builder().tableName(args[0]).build(), jobConf);
OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), jobConf);
Date startTime = new Date();
System.out.println("Job started: " + startTime);
JobClient.runJob(jobConf);
Date end_time = new Date();
System.out.println("Job ended: " + end_time);
System.out.println("The job took " + (end_time.getTime() - startTime.getTime()) / 1000 + " seconds.");
}
}
```

1.8.6.13. Examples of using partitioned tables as input

This topic provides examples of using partitioned tables as input.

Example 1:

```
public static void main(String[] args) throws Exception {
    JobConf job = new JobConf();
    ...
    LinkedHashMap<String, String> input = new LinkedHashMap<String, String>();
    input.put("pt", "123456");
    InputUtils.addTable(TableInfo.builder().tableName("input_table").partSpec(input).build(), job);
    LinkedHashMap<String, String> output = new LinkedHashMap<String, String>();
    output.put("ds", "654321");
    OutputUtils.addTable(TableInfo.builder().tableName("output_table").partSpec(output).build(), job);
    JobClient.runJob(job);
}
```

Example 2:

```
package com.aliyun.odps.mapred.open.example;
...
public static void main(String[] args) throws Exception {
    if (args.length != 2) {
        System.err.println("Usage: WordCount <in_table> <out_table>");
        System.exit(2);
    }
    JobConf job = new JobConf();
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(SumCombiner.class);
    job.setReducerClass(SumReducer.class);
    job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
    job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
    Account account = new AliyunAccount("my_access_id", "my_access_key");
    Odps odps = new Odps(account);
    odps.setEndpoint("odps_endpoint_url");
    odps.setDefaultProject("my_project");
    Table table = odps.tables().get(tblname);
    TableInfoBuilder builder = TableInfo.builder().tableName(tblname);
    for (Partition p : table.getPartitions()) {
        if (applicable(p)) {
            LinkedHashMap<String, String> partSpec = new LinkedHashMap<String, String>();
            for (String key : p.getPartitionSpec().keys()) {
                partSpec.put(key, p.getPartitionSpec().get(key));
            }
            InputUtils.addTable(builder.partSpec(partSpec).build(), conf);
        }
    }
    OutputUtils.addTable(TableInfo.builder().tableName(args[1]).build(), job);
    JobClient.runJob(job);
}
```

Note

- In example 2, the MaxCompute SDK and MapReduce SDK are combined to implement a MapReduce task that reads data from specific partitions.
- The code in the example cannot be compiled for execution. It is only an example of the main function.
- The applicable function is user logic that determines whether the partition can be used as the input of a MapReduce job.

1.8.6.14. Pipeline example

This topic provides a Pipeline example.

Example:

```
package com.aliyun.odps.mapred.open.example;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.Column;
import com.aliyun.odps.OdpsException;
import com.aliyun.odps.OdpsType;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.TableInfo;
```

```

import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.Job;
import com.aliyun.odps.mapred.MapperBase;
import com.aliyun.odps.mapred.ReducerBase;
import com.aliyun.odps.pipeline.Pipeline;
public class WordCountPipelineTest {
    public static class TokenizerMapper extends MapperBase {
        Record word;
        Record one;
        @Override
        public void setup(TaskContext context) throws IOException {
            word = context.createMapOutputKeyRecord();
            one = context.createMapOutputValueRecord();
            one.setBigint(0, 1L);
        }
        @Override
        public void map(long recordNum, Record record, TaskContext context)
            throws IOException {
            for (int i = 0; i < record.getColumnCount(); i++) {
                String[] words = record.get(i).toString().split("\\s+");
                for (String w : words) {
                    word.setString(0, w);
                    context.write(word, one);
                }
            }
        }
    }
    public static class SumReducer extends ReducerBase {
        private Record value;
        @Override
        public void setup(TaskContext context) throws IOException {
            value = context.createOutputValueRecord();
        }
        @Override
        public void reduce(Record key, Iterator<Record> values, TaskContext context)
            throws IOException {
            long count = 0;
            while (values.hasNext()) {
                Record val = values.next();
                count += (Long) val.get(0);
            }
            value.set(0, count);
            context.write(key, value);
        }
    }
    public static class IdentityReducer extends ReducerBase {
        private Record result;
        @Override
        public void setup(TaskContext context) throws IOException {
            result = context.createOutputRecord();
        }
        @Override
        public void reduce(Record key, Iterator<Record> values, TaskContext context)
            throws IOException {
            while (values.hasNext()) {
                result.set(0, key.get(0));
                result.set(1, values.next().get(0));
                context.write(result);
            }
        }
    }
}

```

```
    }  
  }  
}  
public static void main(String[] args) throws OdpsException {  
  if (args.length != 2) {  
    System.err.println("Usage: WordCountPipeline <in_table> <out_table>");  
    System.exit(2);  
  }  
  Job job = new Job();  
  /**  
   * During pipeline construction, if you do not specify OutputKeySortColumns, PartitionColumns, and OutputGroupingColumns for a mapper, the framework uses OutputKey of the mapper as the default values of these parameters.  
   ***/  
  Pipeline pipeline = Pipeline.builder()  
    .addMapper(TokenizerMapper.class)  
    .setOutputKeySchema(  
      new Column[] { new Column("word", OdpsType.STRING) })  
    .setOutputValueSchema(  
      new Column[] { new Column("count", OdpsType.BIGINT) })  
    .setOutputKeySortColumns(new String[] { "word" })  
    .setPartitionColumns(new String[] { "word" })  
    .setOutputGroupingColumns(new String[] { "word" })  
    .addReducer(SumReducer.class)  
    .setOutputKeySchema(  
      new Column[] { new Column("word", OdpsType.STRING) })  
    .setOutputValueSchema(  
      new Column[] { new Column("count", OdpsType.BIGINT)})  
    .addReducer(IdentityReducer.class).createPipeline();  
  // Add the pipeline to JobConf. If you want to configure a combiner, use JobConf.  
  job.setPipeline(pipeline);  
  // Configure the input and output tables.  
  job.addInput(TableInfo.builder().tableName(args[0]).build());  
  job.addOutput(TableInfo.builder().tableName(args[1]).build());  
  // Submit the job and wait for the job to complete.  
  job.submit();  
  job.waitForCompletion();  
  System.exit(job.isSuccessful() == true ? 0 : 1);  
}
```

1.9. MaxCompute Graph

1.9.1. Graph overview

1.9.1.1. Overview

This topic describes the components of MaxCompute Graph and the supported operations.

MaxCompute Graph is a processing framework designed for iterative graph computing. Graph computing jobs use graphs to build models. A graph is a collection of vertices and edges that have values.

MaxCompute Graph allows you to perform the following operations on graphs:

- Change the value of a vertex or edge.
- Add or remove a vertex.

- Add or remove an edge.

Note

- When you edit a vertex or edge, you must use code to maintain the relationships between vertices and edges.
- A directed graph has the following types of edges.
 - Outgoing edge: a directed edge on which the current vertex is the origin.
 - Incoming edge: a directed edge on which the current vertex is the destination.

MaxCompute Graph performs iterations to edit graphs, enable graphs to evolve, and obtain analysis results. Typical applications include the PageRank, single source shortest path (SSSP) algorithm, and k-means clustering algorithm. You can use SDK for Java provided by MaxCompute Graph to compile graph computing programs.

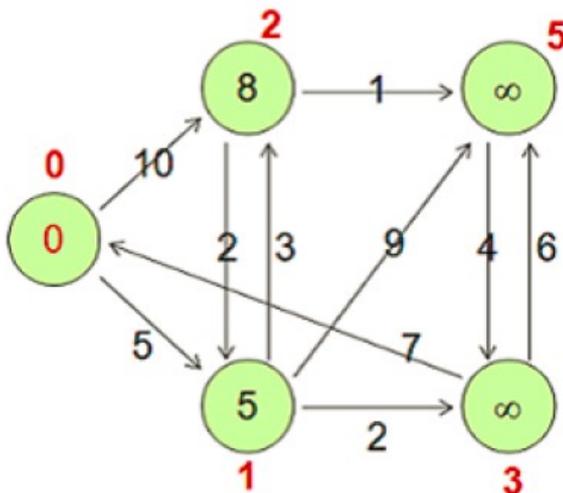
1.9.1.2. Data structure of MaxCompute Graph

This topic describes the data structure of MaxCompute Graph.

MaxCompute Graph processes directed graphs. MaxCompute provides only a two-dimensional table storage structure. Therefore, you must resolve graph data into two-dimensional tables and store them in MaxCompute.

During graph computing and analytics, use custom GraphLoader to convert two-dimensional table data into vertices and edges that are applicable to MaxCompute Graph. You can determine how to resolve graph data into two-dimensional tables based on your business requirements.

The structure of a vertex is <ID, Value, Halted, Edges>. ID specifies the identifier of the vertex. Value specifies the value of the vertex. Halted specifies whether the iteration of the vertex is terminated. Edges specifies all the edges that start from the vertex. The structure of an edge is <DestVertexID, Value>. DestVertexID specifies the identifier of the destination vertex. Value specifies the value of the edge. The following figure shows the data structure of MaxCompute Graph.



The preceding figure is composed of the vertexes listed in the following table.

Vertex	<ID, Value, Halted, Edges>
v0	<0, 0, false, [<1, 5 >, <2, 10 >]>

Vertex	<ID, Value, Halted, Edges>
v1	<1, 5, false, [<2, 3>, <3, 2>, <5, 9>]>
v2	<2, 8, false, [<1, 2>, <5, 1 >]>
v3	<3, Long.MAX_VALUE, false, [<0, 7>, <5, 6>]>
v5	<5, Long.MAX_VALUE, false, [<3, 4 >]>

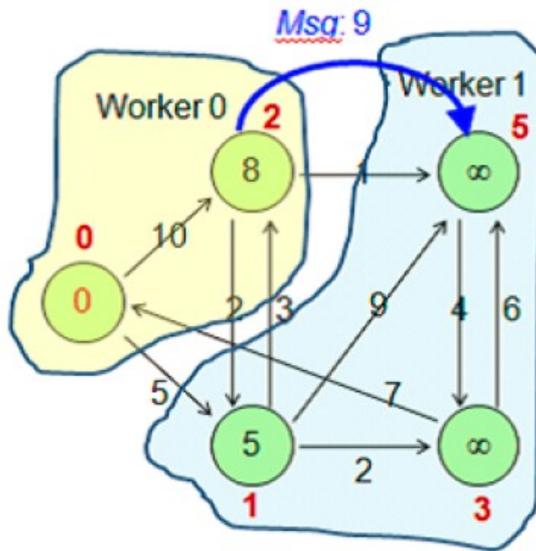
1.9.1.3. Graph logic

1.9.1.3.1. Graph loading

A Graph program performs the following operations: graph loading, iterative computing, and iteration termination. This topic describes the graph loading steps in a Graph program.

Graph loading consists of the following steps:

- Graph loading: MaxCompute Graph calls the custom GraphLoader to resolve the records in the input table into vertices or edges.
- Partitioning: MaxCompute Graph calls the custom Partitioner to partition the vertices and distributes the partitions to the related workers. By default, MaxCompute Graph partitions the vertices based on the hash values of the vertex IDs modulo the number of workers.



In the preceding figure, the number of workers is 2. Vertex 0 and Vertex 2 are distributed to Worker 0 because the result of vertex IDs modulo 2 is 0. Vertex 1, Vertex 3, and Vertex 5 are distributed to Worker 1 because the result of vertex IDs modulo 2 is 1.

1.9.1.3.2. Iterative computing

A Graph program performs the following operations: graph loading, iterative computing, and iteration termination. This topic describes the iterative computing steps in a Graph program.

An iteration is called a superstep. During an iteration, the program traverses all non-halted vertices or all vertices that receive messages, and calls the `compute(ComputeContext context, Iterable messages)` method. For non-halted vertices, the value of the `Halted` parameter is `false`. Halted vertices are automatically activated after they receive messages.

The `compute(ComputeContext context, Iterable messages)` method can be used to perform the following operations:

- Process the messages that are sent by the previous superstep to the current vertex.
- Edit a graph as required:
 - Change the values of vertices or edges.
 - Send messages to some vertices.
 - Add or remove vertices or edges.
- Aggregate information to obtain global information by using the aggregator.
- Set the current vertex to the halted or non-halted state.
- Enable MaxCompute Graph to asynchronously send messages to the related workers in each superstep. The messages are then processed in the next superstep.

1.9.1.3.3. Iteration termination

A Graph program performs the following operations: graph loading, iterative computing, and iteration termination. This topic describes the iteration termination steps in a Graph program.

An iteration ends if one of the following conditions is met:

- All vertices are in the halted state (the value of the `Halted` parameter is `true`), and no new messages are generated.
- The maximum number of iterations is reached.
- The `terminate` method of an aggregator returns `true`.

Example:

```
// 1. load
for each record in input_table { GraphLoader.load();
}
// 2. setup
WorkerComputer.setup();
for each aggr in aggregators { aggr.createStartupValue();
}
for each v in vertices { v.setup();
}
// 3. superstep
for (step = 0; step < max; step++) { for each aggr in aggregators { aggr.createInitialValue();
}
for each v in vertices { v.compute();
}
}
// 4. cleanup
for each v in vertices { v.cleanup();
}
WorkerComputer.cleanup();
```

1.9.1.4. Aggregator mechanism

This topic describes the implementation mechanism and related API operations of Aggregator. It also describes how to apply Aggregator by using k-means clustering.

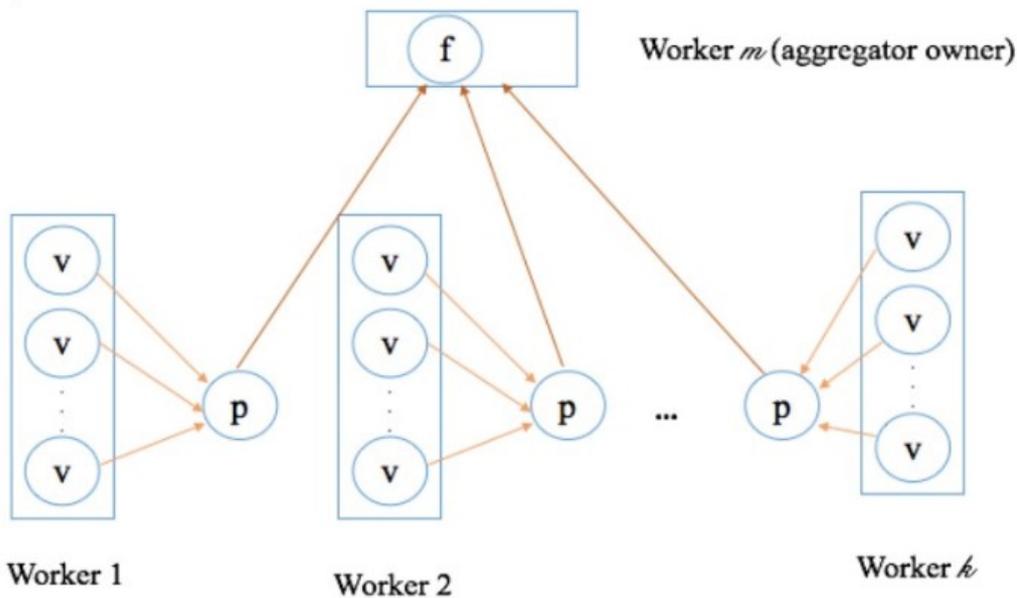
Aggregator is a common feature in MaxCompute Graph jobs and is especially suited for solving machine learning issues. In MaxCompute Graph, Aggregator is used to aggregate and process global information.

Implementation mechanism

The logic of Aggregator is divided into two parts:

- One part is implemented on all workers in distributed mode.
- The other part is only implemented on the worker where Aggregator owner is located in single vertex mode.

Initial values are created and partially aggregated on each worker, and then the partial aggregation results of all workers are sent to the worker where Aggregator owner is located. This worker then aggregates the received partial aggregation objects into a global aggregation result and determines whether to end the iteration. The global aggregation result is distributed to all workers in the next superstep for iteration.



API operations

Aggregator provides five API operations. The following parts describe when to call these API operations and for what purposes.

- createStartupValue(context)

This API operation is performed once on all workers before each superstep starts. It is used to initialize AggregatorValue. In the first superstep (superstep 0), WorkerContext.getLastAggregatedValue() or ComputeContext.getLastAggregatedValue() is called to obtain the initialized AggregatorValue object.

- createInitialValue(context)

This API operation is performed once on all workers when each superstep starts. It is used to initialize AggregatorValue for the current iteration. Generally, WorkerContext.getLastAggregatedValue() is called to obtain the result of the previous iteration, and then partial initialization is implemented.

- aggregate(value, item)

This API operation is also performed on all workers. It is triggered by an explicit call to ComputeContext#aggregate(item), while the preceding two API operations are automatically called by the framework.

This API operation is used to implement partial aggregation. The first parameter value indicates the aggregation result of the worker in the current superstep. The initial value is the object returned by `createInitialValue`. The second parameter is passed in when `ComputeContext#aggregate(item)` is called by using user code. In this API operation, `item` is typically used to update value for aggregation. After all the aggregate operations are performed, the obtained value is the partial aggregation result of the worker. The result is then sent by the framework to the worker where Aggregator or owner is located.

- `merge(value, partial)`

This API operation is performed on the worker where the Aggregator or owner resides. It is used to merge partial aggregation results of workers to obtain the global aggregation object. Similar to `aggregate`, `value` in this API operation indicates the aggregated results, and `partial` indicates objects that you want to aggregate. `partial` is used to update value.

For example, three workers `w0`, `w1`, and `w2` generate partial aggregation results `p0`, `p1`, and `p2`. If `p1`, `p0`, and `p2` are sent in sequence to the worker where the Aggregator or owner resides, the merge operations are performed in the following sequence:

- i. `merge(p1, p0)` is first executed to aggregate `p1` and `p0` as `p1`.
- ii. `merge(p1, p2)` is executed to aggregate `p1` and `p2` as `p1`. `p1` is the global aggregation result in this superstep.

Therefore, if only one worker exists, the merge method is not required. In this case, `merge()` is not called.

- `terminate(context, value)`

After the worker where the Aggregator or owner resides executes `merge()`, the framework calls `terminate(context, value)` to perform the final processing. The second parameter value indicates the global aggregation result obtained by calling `merge()`. The global aggregation result can be further modified in this API operation. After `terminate()` is executed, the framework distributes the global aggregation object to all workers for the next superstep.

If `true` is returned for `terminate()`, iteration is ended for the entire job. Otherwise, the iteration continues. If `true` is returned after convergence is completed, jobs are immediately ended. This applies to machine learning scenarios.

K-means clustering example

This section uses k-means clustering as an example to demonstrate how to use Aggregator.

 **Note** For the complete code, see [Kmeans](#). In this section, the code is resolved and is for reference only.

- `GraphLoader`

`GraphLoader` is used to load an input table and convert it to vertices or edges of a graph. In this example, each row of data in the input table is a sample, each sample constructs a vertex, and vertex values are used to store samples.

A `Writable` class `KmeansValue` is defined as the value type of a vertex.

```
public static class KmeansValue implements Writable {
    DenseVector sample;
    public KmeansValue() {
    }
    public KmeansValue(DenseVector v) {
        this.sample = v;
    }
    @Override
    public void write(DataOutput out) throws IOException {
        writeForDenseVector(out, sample);
    }
    @Override
    public void readFields(DataInput in) throws IOException {
        sample = readFieldsForDenseVector(in);
    }
}
```

A `DenseVector` object is encapsulated in `KmeansValue` to store a sample. The `DenseVector` type stems from [matrix-toolkits-java](#). `writeForDenseVector()` and `readFieldsForDenseVector()` are used for serialization and deserialization. For more information, see the complete code.

Custom `KmeansReader` code:

```
public static class KmeansReader extends
    GraphLoader<LongWritable, KmeansValue, NullWritable, NullWritable> {
    @Override
    public void load(
        LongWritable recordNum,
        WritableRecord record,
        MutationContext<LongWritable, KmeansValue, NullWritable, NullWritable> context)
        throws IOException {
        KmeansVertex v = new KmeansVertex();
        v.setId(recordNum);
        int n = record.size();
        DenseVector dv = new DenseVector(n);
        for (int i = 0; i < n; i++) {
            dv.set(i, ((DoubleWritable)record.get(i)).get());
        }
        v.setValue(new KmeansValue(dv));
        context.addVertexRequest(v);
    }
}
```

In `KmeansReader`, a vertex is created when each row of data (a record) is read. `recordNum` is used as the vertex ID, and the record content is converted to a `DenseVector` object and encapsulated in `VertexValue`.

- Vertex

Custom `KmeansVertex` code:

```
public static class KmeansVertex extends
Vertex<LongWritable, KmeansValue, NullWritable, NullWritable> {
    @Override
    public void compute(
        ComputeContext<LongWritable, KmeansValue, NullWritable, NullWritable> context,
        Iterable<NullWritable> messages) throws IOException {
        context.aggregate(getValue());
    }
}
```

The logic of the preceding code is to implement partial aggregation for samples maintained for each iteration. For more information about the logic, see the implementation of `Aggregator` in the following section.

- **Aggregator**

The main logic of k-means is concentrated on `Aggregator`. Custom `KmeansAggrValue` is used to maintain the content you want to aggregate and distribute.

```
public static class KmeansAggrValue implements Writable {
    DenseMatrix centroids;
    DenseMatrix sums; // used to recalculate new centroids
    DenseVector counts; // used to recalculate new centroids
    @Override
    public void write(DataOutput out) throws IOException {
        writeForDenseDenseMatrix(out, centroids);
        writeForDenseDenseMatrix(out, sums);
        writeForDenseVector(out, counts);
    }
    @Override
    public void readFields(DataInput in) throws IOException {
        centroids = readFieldsForDenseMatrix(in);
        sums = readFieldsForDenseMatrix(in);
        counts = readFieldsForDenseVector(in);
    }
}
```

In the preceding code, three objects are maintained in `KmeansAggrValue`:

- `centroids`: indicates the existing K centers. If the sample is m -dimensional, `centroids` is a matrix of $K \times m$.
- `sums`: indicates a matrix of the same size as `centroids`. Each element records the sum of a specific dimension of the sample that is closest to a specific center. For example, `sums(i,j)` indicates the sum of dimension j of the sample that is closest to center i .
- `counts` is a K -dimensional vector. It records the number of samples closest to each center. `counts` is used with `sums` to calculate a new center, which is the main content to be aggregated.

`KmeansAggregator` is a custom `Aggregator` implementation class. The implementation is described below in order of the preceding API operations:

- i. Implementation of `createStartupValue()`

```
public static class KmeansAggregator extends Aggregator<KmeansAggrValue> {
    public KmeansAggrValue createStartupValue(WorkerContext context) throws IOException {
        KmeansAggrValue av = new KmeansAggrValue();
        byte[] centers = context.readCacheFile("centers");
        String lines[] = new String(centers).split("\n");
        int rows = lines.length;
        int cols = lines[0].split(",").length; // assumption rows >= 1
        av.centroids = new DenseMatrix(rows, cols);
        av.sums = new DenseMatrix(rows, cols);
        av.sums.zero();
        av.counts = new DenseVector(rows);
        av.counts.zero();
        for (int i = 0; i < lines.length; i++) {
            String[] ss = lines[i].split(",");
            for (int j = 0; j < ss.length; j++) {
                av.centroids.set(i, j, Double.valueOf(ss[j]));
            }
        }
        return av;
    }
}
```

This method initializes a `KmeansAggrValue` object, reads the initial center from the centers file, and assigns a value to centroids. The initial values of sums and counts are 0.

ii. Implementation of `createInitialValue()`

```
@Override
public KmeansAggrValue createInitialValue(WorkerContext context)
    throws IOException {
    KmeansAggrValue av = (KmeansAggrValue)context.getLastAggregatedValue(0);
    // reset for next iteration
    av.sums.zero();
    av.counts.zero();
    return av;
}
```

This method first obtains `KmeansAggrValue` of the previous iteration and clears the values of sums and counts. Only the centroids value of the previous iteration is retained.

iii. Implementation of `aggregate()`

```
@Override
public void aggregate(KmeansAggrValue value, Object item)
    throws IOException {
    DenseVector sample = ((KmeansValue)item).sample;
    // find the nearest centroid
    int min = findNearestCentroid(value.centroids, sample);
    // update sum and count
    for (int i = 0; i < sample.size(); i++) {
        value.sums.add(min, i, sample.get(i));
    }
    value.counts.add(min, 1.0d);
}
```

This method calls `findNearestCentroid()` to find the index of the center closest to the sample item, uses sums to add up all dimensions, and increments the value of counts by 1.

The preceding three methods are executed on all workers to implement partial aggregation. The following methods can be used to implement global aggregation on the worker where the Aggregator owner resides:

i. Implementation of merge()

```
@Override
public void merge(KmeansAggrValue value, KmeansAggrValue partial)
    throws IOException {
    value.sums.add(partial.sums);
    value.counts.add(partial.counts);
}
```

In the preceding example, the implementation logic of merge is to add the values of sums and counts aggregated by each worker.

ii. Implementation of terminate()

```
@Override
public boolean terminate(WorkerContext context, KmeansAggrValue value)
    throws IOException {
    // Calculate the new means to be the centroids (original sums)
    DenseMatrix newCentroids = calculateNewCentroids(value.sums, value.counts, value.centroids);
    // print old centroids and new centroids for debugging
    System.out.println("\nsuperstep: " + context.getSuperstep() +
        "\nold centroid:\n" + value.centroids + " new centroid:\n" + newCentroids);
    boolean converged = isConverged(newCentroids, value.centroids, 0.05d);
    System.out.println("superstep: " + context.getSuperstep() + "/"
        + (context.getMaxIteration() - 1) + " converged: " + converged);
    if (converged || context.getSuperstep() == context.getMaxIteration() - 1) {
        // converged or reach max iteration, output centroids
        for (int i = 0; i < newCentroids.numRows(); i++) {
            Writable[] centroid = new Writable[newCentroids.numColumns()];
            for (int j = 0; j < newCentroids.numColumns(); j++) {
                centroid[j] = new DoubleWritable(newCentroids.get(i, j));
            }
            context.write(centroid);
        }
        // true means to terminate iteration
        return true;
    }
    // update centroids
    value.centroids.set(newCentroids);
    // false means to continue iteration
    return false;
}
```

In the preceding example, terminate() calls calculateNewCentroids() based on sums and counts to calculate the average value and obtain a new center. Then, isConverged() is called to check whether the center is converged based on the Euclidean distance between the new and old centers. If the number of convergences or iterations reaches the upper limit, the new center is generated, and true is returned to end the iteration. Otherwise, the center is updated, and false is returned to continue the iteration.

iii. main method

The main method is used to construct GraphJob, configure related settings, and submit a job.

```
public static void main(String[] args) throws IOException {
    if (args.length < 2)
        printUsage();
    GraphJob job = new GraphJob();
    job.setGraphLoaderClass(KmeansReader.class);
    job.setRuntimePartitioning(false);
    job.setVertexClass(KmeansVertex.class);
    job.setAggregatorClass(KmeansAggregator.class);
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addOutput(TableInfo.builder().tableName(args[1]).build());
    // default max iteration is 30
    job.setMaxIteration(30);
    if (args.length >= 3)
        job.setMaxIteration(Integer.parseInt(args[2]));
    long start = System.currentTimeMillis();
    job.run();
    System.out.println("Job Finished in "
        + (System.currentTimeMillis() - start) / 1000.0 + " seconds");
}
```

 **Note** If `job.setRuntimePartitioning` is set to `false`, data loaded by each worker is not partitioned based on the partitioner. Data is loaded and maintained by the same worker.

Summary

Basic steps to implement Aggregator:

1. When each worker starts, it executes `createStartUpValue` to create `AggregatorValue`.
2. Before each iteration starts, each worker executes `createInitialValue` to initialize `AggregatorValue` for the iteration.
3. In an iteration, each vertex uses `context.aggregate()` to call `aggregate()` to implement partial iteration in the worker.
4. Each worker sends the partial iteration result to the worker where the Aggregator owner resides.
5. The worker where the Aggregator owner resides executes `merge` multiple times to implement global aggregation.
6. The worker where the Aggregator owner resides executes `terminate` to process the global aggregation result and determines whether to end the iteration.

1.9.2. Graph feature overview

1.9.2.1. Run a job

This topic describes how to run a MaxCompute Graph job.

The MaxCompute client provides a JAR command for you to run MaxCompute Graph jobs. This command is used in the same way as the JAR command in MapReduce.

Syntax:

```
Usage: jar [<GENERIC_OPTIONS>] <MAIN_CLASS> [ARGS]
  -conf <configuration_file>  Specify an application configuration file
  -classpath <local_file_list> classpaths used to run mainClass
  -D <name>=<value>           Property value pair, which will be used to run mainClass
  -local                       Run job in local mode
  -resources <resource_name_list> file/table resources used in graph, separate by comma
```

The following table describes the parameters in this command.

Parameters

Parameter	Description
-conf <configuration file>	The JobConf configuration file.
-classpath <local_file_list>	The classpath used to run a job in local mode. This parameter specifies the local paths of the JAR package where the Main function is located. The paths include both the relative and absolute paths.
-D <prop_name>=<prop_value>	The Java property of <mainClass> for local execution. You can specify multiple properties.
-local	Specifies that the Graph jobs run in local mode. It is used for program debugging.
-resources <resource_name_list>	Declares the resources used for running the Graph job. In most cases, you must specify the names of the resources that the Graph job uses in resource_name_list. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note If you read other MaxCompute resources when you run the Graph job, you must add the names of those resources to <resource_name_list>. Multiple resources must be separated by commas (.). If you use cross-project resources, add PROJECT_NAME/resources/ before <resource_name_list>. Example: -resources otherproject/resources/resfile.</p> </div>

 **Note** The preceding optional parameters are included in <GENERIC_OPTIONS>.

You can directly run the main function in the Graph job to submit the job to MaxCompute, instead of submitting the job by using the MaxCompute client. The PageRank algorithm is used in the following example:

```
public static void main(String[] args) throws IOException {
    if (args.length < 2)
        printUsage();
    GraphJob job = new GraphJob();
    job.setGraphLoaderClass(PageRankVertexReader.class);
    job.setVertexClass(PageRankVertex.class);
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addOutput(TableInfo.builder().tableName(args[1]).build());
    // Add the resources used in the job to the cache resource. These resources correspond to those specified by -resources and -libjars in the jar command.
    job.addCacheResource("mapreduce-examples.jar");
    // Add the used JAR file and other files to the class cache resource. These files correspond to those specified by -libjars in the JAR command.
    job.addCacheResourceToClassPath("mapreduce-examples.jar");
    // Set the configuration item that corresponds to odps_config.ini in the client. Replace it with the actual one in your configuration file.
    Account account = new AliyunAccount(accessId, accessKey);
    Odps odps = new Odps(account);
    odps.setDefaultProject(project);
    odps.setEndpoint(endpoint);
    SessionState.get().setOdps(odps);
    SessionState.get().setLocalRun(false); // default max iteration is 30
    job.setMaxIteration(30);
    if (args.length >= 3)
        job.setMaxIteration(Integer.parseInt(args[2]));
    long startTime = System.currentTimeMillis(); job.run();
    System.out.println("Job Finished in "
        + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
```

1.9.2.2. Input and output

This topic describes how to define the input and output of a MaxCompute Graph job.

The input and output of MaxCompute Graph jobs must be tables. You cannot customize the input or output format.

Multiple tables and partitions can be used as input. Example of job input definition:

```
GraphJob job = new GraphJob();
job.addInput(TableInfo.builder().tableName("tblName").build());
// Tables are used as input.
job.addInput(TableInfo.builder().tableName("tblName").partSpec("pt1=a/pt2=b").build());
// Partitions are used as input.
job.addInput(TableInfo.builder().tableName("tblName").partSpec("pt1=a/pt2=b").build(), new String[]{"col2", "col0"});
// Read only the col2 and col0 columns of the input table. Use record.get(0) to obtain col2 in the load() method of GraphLoader. Both are read in the same sequence.
```

Note

- Partition filtering is not supported.
- The addInput framework reads records from the input table and transfers the records to the user-defined GraphLoader to load graph data.

Multiple tables and partitions can be used as output, and each of them must be marked with a label. Example of job output definition:

```
GraphJob job = new GraphJob();
job.addOutput(TableInfo.builder().tableName("table_name").partSpec("pt1=a/pt2=b").build());
// If the output table is a partitioned table, the last level of partitions must be provided.
job.addOutput(TableInfo.builder().tableName("table_name").partSpec("pt1=a/pt2=b").label("output1").build(), true);
// true indicates that the code overwrites partitions specified by tableinfo. The value true is similar to the INSERT OVERWRITE statement. The value false is similar to the INSERT INTO statement.
```

 **Note** A Graph job can use the `write()` method of `WorkContext` to write data to an output table during runtime. If you want to write data to multiple tables, you must mark each table with a label, such as `output1` in the preceding example.

1.9.2.3. Read data from resources

1.9.2.3.1. Use GraphJob to specify resources to be read

This topic describes how to use `GraphJob` to read resources in MaxCompute Graph.

In addition to the `JAR` command, you can use the following two methods of `GraphJob` to specify the resources to be read by Graph:

```
void addCacheResources(String resourceNames)
void addCacheResourcesToClassPath(String resourceNames)
```

1.9.2.3.2. Use resources in Graph

This topic describes how to use `WorkerContext` to read resources in MaxCompute Graph.

You can use a `WorkerContext` object to read resources in MaxCompute Graph.

```
public byte[] readCacheFile(String resourceName) throws IOException;
public Iterable<byte[]> readCacheArchive(String resourceName) throws IOException;
public Iterable<byte[]> readCacheArchive(String resourceName, String relativePath) throws IOException;
public Iterable<WritableRecord> readResourceTable(String resourceName);
public BufferedInputStream readCacheFileAsStream(String resourceName) throws IOException;
public Iterable<BufferedInputStream> readCacheArchiveAsStream(String resourceName) throws IOException;
public Iterable<BufferedInputStream> readCacheArchiveAsStream(String resourceName, String relativePath) throws IOException;
```

 **Note**

- You can use the `setup()` method of `WorkerComputer` to read resources and save the resources in `WorkerValue`. Then, you can use `getWorkerValue` to obtain the resources.
- If you want to read resources and process them at the same time, you can use the preceding stream APIs. This method reduces memory consumption.

1.9.3. Graph SDK

This topic describes the commonly used classes of Graph SDK.

Class	Description
GraphJob	Defines, submits, and manages a MaxCompute Graph job. The class inherits JobConf.
Vertex	Defines a vertex in a graph. A vertex has the following properties: id, value, halted, and edges. You can specify this class by using the setVertexClass method of GraphJob.
Edge	Defines an edge in a graph. An edge has the following properties: destVertexId and value. MaxCompute Graph uses the adjacency list as the data structure. The outgoing edges of a vertex are specified by the edges property of the vertex.
GraphLoader	Loads a graph. You can specify this class by using the setGraphLoaderClass method of GraphJob.
VertexResolver	Customizes the logic for handling conflicts during graph topology modification. You can specify this class by using the setLoadingVertexResolverClass and setComputingVertexResolverClass methods of GraphJob. The setLoadingVertexResolverClass method specifies the class that is used during graph loading, whereas the setComputingVertexResolverClass method specifies the class that is used during iterative computing.
Partitioner	Specifies how to distribute vertices to workers. This way, computing can be performed by multiple workers at the same time. You can specify this class by using the setPartitionerClass method of GraphJob. By default, the HashPartitioner class is used. HashPartitioner computes the hash value of a vertex ID and divides the hash value by the number of workers to obtain the remainder of the hash value. This remainder specifies the worker to which the vertex is distributed.
WorkerComputer	Customizes the operations to perform when a worker starts and stops. You can specify this class by using the setWorkerComputerClass method of GraphJob.
Aggregator	Processes and summarizes global information. You can specify one or more Aggregator classes by using the setAggregatorClass(Class...) method of GraphJob.
Combiner	Summarizes the output records. You can specify this class by using the setCombinerClass method of GraphJob.
Counters	Defines a counter that is used for counting workers. In the operating logic of a job, you can use the WorkerContext class to obtain the counter and count workers. The framework then automatically calculates the sum of the workers.
WorkerContext	Defines the context that encapsulates the features provided by the framework, such as the features that allow workers to modify the graph topology, send messages, write results, and read resources.

1.9.4. Development and debugging

1.9.4.1. Development process

This topic describes the development process of MaxCompute Graph.

Graph development plug-ins are unavailable in MaxCompute. You can use Eclipse to develop MaxCompute Graph programs. We recommend that you perform the following steps to develop a MaxCompute Graph program:

1. Compile Graph code and perform local debugging to test basic functions.

2. Perform cluster debugging to verify the result.

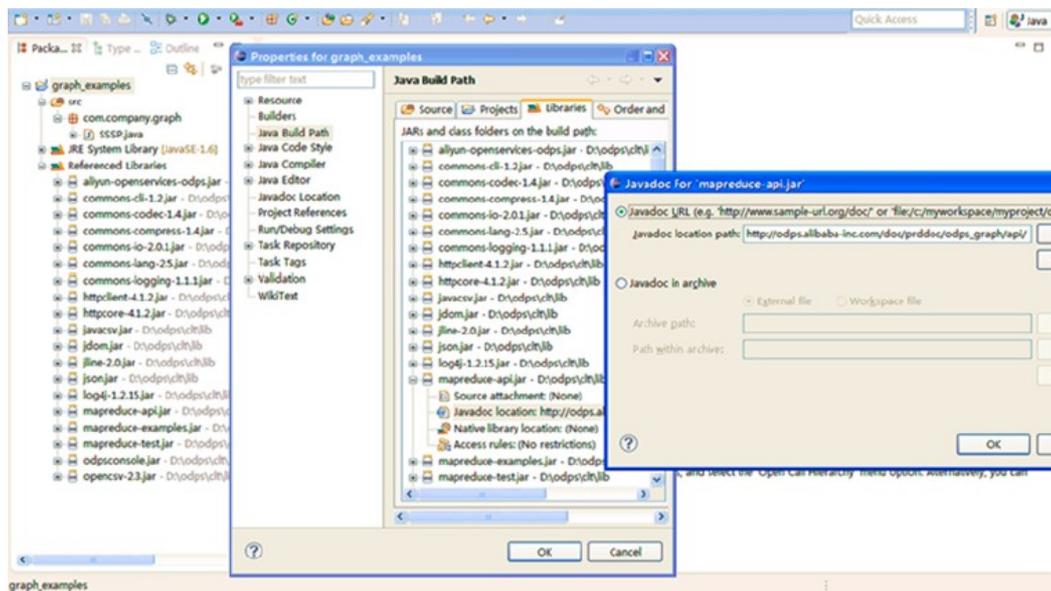
1.9.4.2. Development example

This topic uses the single source shortest path (SSSP) algorithm as an example to describe how to use Eclipse to develop and debug a MaxCompute Graph program.

Procedure

1. Create a Java project named graph_examples.
2. Add the JAR package in the lib directory on the MaxCompute client to **Java Build Path** of the Eclipse project.

The following figure shows a configured Eclipse project.



3. Develop a MaxCompute Graph program.
In the actual development process, an example program, such as SSSP, is first copied and then modified. In this example, only the package path is changed to *package com.aliyun.odps.graph.example*.
4. Compile and package the code. In an Eclipse environment, right-click the source code directory (the src directory in the preceding figure) and choose **Export > Java > JAR file** to generate a JAR package. Then, select the path to store the JAR package, such as *D:\odps\clt\odps-graph-example-sssp.jar*.
5. Use the MaxCompute client to run SSSP. For more information, see [Compile and run a Graph job](#).

1.9.4.3. Perform debugging on the local computer

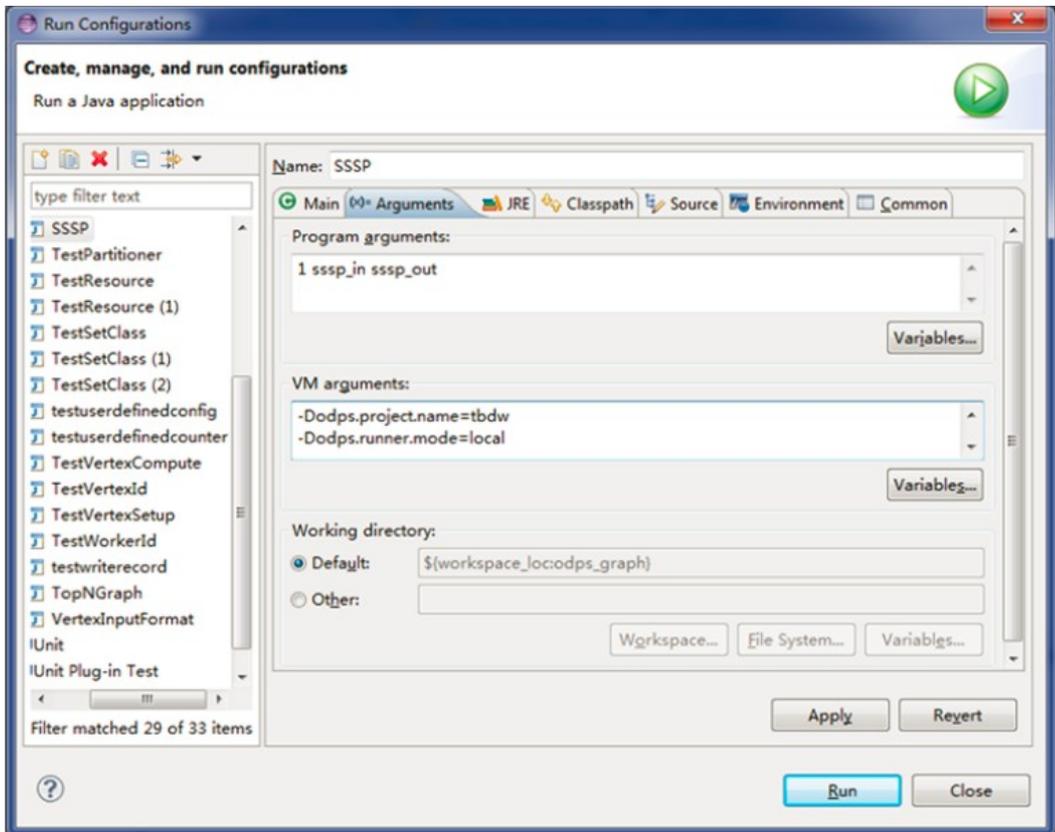
MaxCompute Graph supports the local debugging mode. This topic describes how to use Eclipse for breakpoint debugging.

Procedure

1. Download a [Maven package](#) named odps-graph-local.
2. Select the Eclipse project, right-click the main program file that contains the main function of a Graph job, and choose **Run As > Run Configurations** to configure parameters.
3. On the **Arguments** tab, set Program arguments to `1 sssp_in sssp_out` as the input parameter of the main program.

4. On the **Arguments** tab, set VM arguments to the following content:

```
-Dodps.runner.mode=local  
-Dodps.project.name=<project.name>  
-Dodps.end.point=<end.point>  
-Dodps.access.id=<access.id>  
-Dodps.access.key=<access.key>
```



5. In local mode in which `odps.end.point` is not specified, create the `sssp_in` and `sssp_out` tables in the warehouse directory and add the following data to the `sssp_in` table:

```
1,"2:2,3:1,4:4"  
2,"1:2,3:2,4:1"  
3,"1:1,2:2,5:1"  
4,"1:4,2:1,5:1"  
5,"3:1,4:1"
```

Note For more information about the warehouse directory, see [Run MapReduce tasks locally](#).

6. Click **Run** to run SSSP on the local machine.

Note

Configure the parameters based on the settings in `conf/odps_config.ini` on the MaxCompute client. The preceding parameters are commonly used. Descriptions of other parameters:

- `odps.runner.mode`: The value is `local`. This parameter is required for the local debugging feature.
- `odps.project.name`: specifies the current project. This parameter is required.
- `odps.end.point`: specifies the endpoint of MaxCompute. This parameter is optional. If this parameter is not specified, metadata and data of tables or resources are only read from the warehouse directory. An exception is reported if such data does not exist in the directory. If this parameter is specified, metadata and data are read from the warehouse directory first and then from the remote MaxCompute server if such data does not exist in the directory.
- `odps.access.id`: specifies the AccessKey ID used to access MaxCompute. This parameter is valid only if `odps.end.point` is specified.
- `odps.access.key`: specifies the AccessKey secret used to access MaxCompute. This parameter is valid only if `odps.end.point` is specified.
- `odps.cache.resources`: specifies the resources you want to use. This parameter functions the same as `-resources` of the `jar` command.
- `odps.local.warehouse`: specifies the local path to warehouse. The default value is `./warehouse`.

Output of the local SSSP debugging in Eclipse:

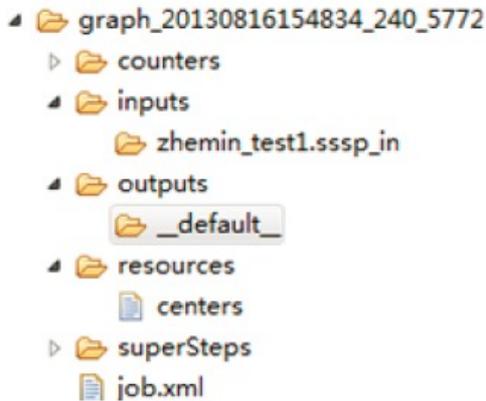
```
Counters: 3
com.aliyun.odps.graph.local.COUNTER
TASK_INPUT_BYTE=211
TASK_INPUT_RECORD=5
TASK_OUTPUT_BYTE=161
TASK_OUTPUT_RECORD=5
graph task finish
```

Notice In this example, the `sssp_in` and `sssp_out` tables must exist in the local warehouse directory. For more information about the `sssp_in` and `sssp_out` tables, see [Compile and run a Graph job](#).

1.9.4.4. Temporary directory of a local job

This topic describes the temporary directory created when MaxCompute Graph runs a local debugging job.

Each time MaxCompute Graph runs a local debugging job, it creates a temporary directory in the Eclipse project directory, as shown in the following figure.



The temporary directory of a local Graph job contains the following directories and files:

- **counters:** stores the counter information that is generated during the job running.
- **inputs:** stores the input data of the job. Input data is read from the local warehouse directory first. If no data is available, MaxCompute SDK reads data from the server if `odps.end.point` is specified. The `-Dodps.mapred.local.record.limit` parameter specifies the number of records that can be read during each input operation. The default value is 10. The maximum value is 10000.
- **outputs:** stores the output data of the job. If an output table exists in the local warehouse, the results in outputs will overwrite data in that table after the job is executed.
- **resources:** stores the resources that are used by the job. Similar to input data, resource data is read from the local warehouse directory first. If no data is available, MaxCompute SDK reads data from the server if `odps.end.point` is specified.
- **job.xml:** stores job configurations.
- **superstep:** stores persistent messages from each iteration.

Note If you want to produce detailed logs during local debugging, place the log4j configuration file named `log4j.properties_odps_graph_cluster_debug` in the `src` directory.

1.9.4.5. Cluster debugging

This topic describes how to submit a job to a cluster for testing.

After local debugging, you can submit the job to a cluster for testing.

1. Configure the MaxCompute client.
2. Run the `add jar /path/work.jar -f;` command to update the JAR package.
3. Run a jar command to run a job. Then, check the operational log and command output.

Note For more information about how to run a Graph job in a cluster, see [Compile and run a Graph job](#).

1.9.4.6. Performance optimization

This topic describes how to configure job parameters to optimize the performance of Graph.

The following table describes the job parameters that affect the performance of Graph.

Parameter	Description
setSplitSize(long)	The split size of an input table. The unit is MB. The value must be greater than 0. The default value is 64.
setNumWorkers(int)	The number of workers for a job. The value ranges from 1 to 1000. The default value is 1. The number of workers is determined by the input bytes of the job and split size.
setWorkerCPU(int)	The number of CPU resources for a map job. 100 resources are equivalent to one CPU core. The value ranges from 50 to 800. The default value is 200.
setWorkerMemory(int)	The size of memory resources for a map job. The unit is MB. The value ranges from 256 to 12288. The default value is 4096.
setMaxIteration(int)	The maximum number of iterations. The default value is -1. If the value is less than or equal to 0, the job does not stop when the maximum number of iterations is reached.
setJobPriority(int)	The job priority. The value ranges from 0 to 9. The default value is 9. A greater value indicates a lower priority.

We recommend that you optimize the performance by using one or more of the following methods:

1. Use `setNumWorkers` to increase the number of workers.
2. Use `setSplitSize` to reduce the split size and increase the data loading speed.
3. Increase the CPU or memory resources for workers.
4. Set the maximum number of iterations. For applications that do not require high precision, you can reduce the number of iterations to accelerate the execution process.

`setNumWorkers` and `setSplitSize` can be used together to accelerate data loading. Assume that the value of `setNumWorkers` equals that of `workerNum`, the value of `setSplitSize` equals that of `splitSize`, and the total number of input bytes is the value of `inputSize`. The number of split data records is calculated by using the following formula: $\text{splitNum} = \text{inputSize} / \text{splitSize}$. Relationship between `workerNum` and `splitNum`:

- If the value of `splitNum` is equal to that of `workerNum`, each worker loads one split data record.
- If the value of `splitNum` is greater than that of `workerNum`, each worker loads one or more split data records.
- If the value of `splitNum` is less than that of `workerNum`, each worker uploads one or no split data record.

Therefore, you can adjust the settings of `workerNum` and `splitNum` to obtain a suitable data loading speed. In the first two cases, data is loaded faster. In the iteration phase, you need only to adjust the setting of `workerNum`. If you set runtime partitioning to `False`, we recommend that you use `setSplitSize` to adjust the number of workers or make sure that the conditions in the first two cases are met. In the third case, some workers do not load split data records. Therefore, you can insert `set odps.graph.split.size=<m>; set odps.graph.worker.num=<n>;` before the JAR command to achieve the same effect as `setNumWorkers` and `setSplitSize`.

Another common performance issue is data skew. As indicated by counters, some workers process much more split data records or edges than other workers.

Data skew occurs when the number of split data records, edges, or messages that correspond to some keys is much greater than that of other keys. These keys are processed by a small number of workers, which results in longer runtime. To address this issue, use one or more of the following methods:

- Use a combiner to aggregate the messages of the split data records that correspond to the keys to reduce the number of messages generated.

Developers can define a combiner to reduce the memory and network traffic consumed by message storage, which shortens the job execution duration.

- Improve the business logic.

If the data volume is large, reading data in a disk may take up the processing time. Therefore, you can reduce the data bytes to be read to increase the overall throughput. This improves job performance. To reduce data bytes, use one of the following methods:

- Reduce data input: For some decision-making applications, processing sampled data affects only the precision of the results, not the overall accuracy. In this case, you can perform special data sampling and import the data to the input table for processing.
- Avoid reading fields that are not used: The `TableInfo` class of the MaxCompute Graph framework supports reading specific columns that are transferred by using column name arrays, rather than reading the entire table or partition. This reduces the input data volume and improves job performance.

1.9.4.7. Built-in JAR packages

This topic describes the built-in JAR packages that are loaded to a Java Virtual Machine (JVM) by default.

By default, the following JAR packages are loaded to a JVM that runs Graph programs. You do not need to manually upload these resources or use `-libjars` to specify them in a command:

- commons-codec-1.3.jar
- commons-io-2.0.1.jar
- commons-lang-2.5.jar
- commons-logging-1.0.4.jar
- commons-logging-api-1.0.4.jar
- guava-14.0.jar
- json.jar
- log4j-1.2.15.jar
- slf4j-api-1.4.3.jar
- slf4j-log4j12-1.4.3.jar
- xmlenc-0.52.jar

 **Note** In the classpath of the JVM, the preceding built-in JAR packages are placed before your JAR packages, which may result in a version conflict. For example, your program calls a specific class function in commons-codec-1.5.jar, but the function is not included in commons-codec-1.3.jar. In this case, you can call a similar function in commons-codec-1.3.jar or wait until MaxCompute is updated to the required version.

1.9.5. Limits

This topic describes the limits of MaxCompute Graph.

- Each job can reference a maximum 256 resources. Each table or archive is considered as one unit.
- The total size of resources referenced by a job cannot exceed 512 MB.
- The number of the inputs of a job cannot exceed 1,024. The number of input tables cannot exceed 64. The number of the outputs of a job cannot exceed 256.
- Labels that are specified for outputs cannot be null or empty strings. A label cannot exceed 256 characters in length and can contain only letters, digits, underscores (`_`), number signs (`#`), periods (`.`), and hyphens (`-`).
- The number of custom counters in a job cannot exceed 64. group name and counter name of counters

cannot contain number signs (#). The total length of the two names cannot exceed 100 characters.

- The number of workers for a job is calculated by the framework. The maximum number of workers is 1,000. If the number of workers exceeds this value, an error is reported.
- By default, a worker consumes 200 CPU resources. 100 resources are equivalent to one CPU core. The number of CPU resources consumed by a worker ranges from 50 to 800.
- By default, a worker consumes 4,096 MB memory. The memory consumed by a worker ranges from 256 MB to 12,288 MB.
- A worker can repeatedly read a resource for a maximum of 64 times.
- The default value of split size is 64 MB. You can set the value based on your requirements. The value of this parameter must be greater than 0 and less than or equal to the result of the $9223372036854775807 \gg 20$ operation.
- GraphLoader, Vertex, and Aggregator in MaxCompute Graph are limited by the Java sandbox when they are run in a cluster. However, the main program of Graph jobs is not limited by the Java sandbox. For more information, see [Java sandbox limits](#).

1.9.6. Sample programs

1.9.6.1. SSSP

This topic provides an example of Single Source Shortest Path (SSSP).

Dijkstra's algorithm is a common algorithm that is used to calculate the SSSP in a directed graph.

How the Dijkstra's algorithm works:

- Initialization: The path from s to s is 0 ($d[s] = 0$), and the path from u to s is infinite ($d[u] = \infty$).
- Iteration: If an edge from u to v exists, the shortest path from s to v is updated to $d[v] = \min(d[v], d[u] + \text{weight}(u, v))$. The iteration does not end until the paths from all vertices to s do not change.

 **Note** Shortest path: For a weighted directed graph $G = (V, E)$, multiple paths are available from source vertex s to sink vertex v . The path with the smallest sum of edge weights is called the shortest path from s to v .

The working mode of the algorithm shows that the algorithm is suitable for MaxCompute Graph. Each vertex maintains the current shortest path to the source vertex. If the path changes, the new path is added with the edge weight, and a message is sent to notify adjacent vertices. In the next iteration, the adjacent vertices update the shortest paths based on the received message. If the shortest path between each vertex and the source vertex does not change, the iteration ends.

Example:

```
import java.io.IOException;
import com.aliyun.odps.io.WritableRecord;
import com.aliyun.odps.graph.Combiner;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.Edge;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.data.TableInfo;
public class SSSP {
    public static final String START_VERTEX = "sssp.start.vertex.id";
```

```
/**Define SSSPVertex, where:
 * The vertex value indicates the current shortest path from this vertex to the source vertex startVertexId.
 * The compute() method uses the iteration formula  $d[v] = \min(d[v], d[u] + \text{weight}(u, v))$  to update the vertex value.
 * The cleanup() method writes the vertex and its shortest path to the source vertex to the result table.
 **/
public static class SSSPVertex extends
Vertex<LongWritable, LongWritable, LongWritable, LongWritable> {
private static long startVertexId = -1;
public SSSPVertex() {
this.setValue(new LongWritable(Long.MAX_VALUE));
}
public boolean isStartVertex(
ComputeContext<LongWritable, LongWritable, LongWritable, LongWritable> context) {
if (startVertexId == -1) {
String s = context.getConfiguration().get(START_VERTEX);
startVertexId = Long.parseLong(s);
}
return getId().get() == startVertexId;
}
@Override
public void compute(
ComputeContext<LongWritable, LongWritable, LongWritable, LongWritable> context, Iterable<LongWritable> mes
sages) throws IOException {
long minDist = isStartVertex(context) ? 0 : Integer.MAX_VALUE;
for (LongWritable msg : messages) { if (msg.get() < minDist) {
minDist = msg.get();
}
}
if (minDist < this.getValue().get()) {
this.setValue(new LongWritable(minDist));
if (hasEdges()) {
for (Edge<LongWritable, LongWritable> e : this.getEdges()) {
context.sendMessage(e.getDestVertexId(), new LongWritable(minDist + e.getValue().get()));
}
}
} else {
voteToHalt();
// If the vertex value does not change, voteToHalt() is called to notify the framework that this vertex enters the halt
ed state. The calculation ends when all vertices enter the halted state.
}
}
@Override
public void cleanup(
WorkerContext<LongWritable, LongWritable, LongWritable, LongWritable> context) throws IOException {
context.write(getId(), getValue());
}
}
/** Define MinLongCombiner and combine messages sent to the same vertex to optimize performance and reduce
memory usage. **/
public static class MinLongCombiner extends
Combiner<LongWritable, LongWritable> {
@Override
public void combine(LongWritable vertexId, LongWritable combinedMessage, LongWritable messageToCombine) t
hrows IOException {
if (combinedMessage.get() > messageToCombine.get()) {
combinedMessage.set(messageToCombine.get());
}
}
}
}
```

```

,
}
/** Define the SSSPVertexReader class, load a graph, and parse each record in the table into a vertex. The first column of the record is the vertex ID, and the second column stores all edge sets starting from the vertex, such as 2:2,3:1,4:4. */
public static class SSSPVertexReader extends
GraphLoader<LongWritable, LongWritable, LongWritable, LongWritable> {
@Override
public void load(LongWritable recordNum, WritableRecord record,
MutationContext<LongWritable, LongWritable, LongWritable, LongWritable> context) throws IOException {
SSSPVertex vertex = new SSSPVertex();
vertex.setID((LongWritable) record.get(0));
String[] edges = record.get(1).toString().split(",");
for (int i = 0; i < edges.length; i++) {
String[] ss = edges[i].split(":");
vertex.addEdge(new LongWritable(Long.parseLong(ss[0])), new LongWritable(Long.parseLong(ss[1])));
}
context.addVertexRequest(vertex);
}
}
public static void main(String[] args) throws IOException { if (args.length < 2) {
System.out.println("Usage: <startnode> <input> <output>");
System.exit(-1);
}
GraphJob job = new GraphJob();
// Define GraphJob, specify the implementation of Vertex, GraphLoader, and Combiner, and specify input and output tables.
job.setGraphLoaderClass(SSSPVertexReader.class);
job.setVertexClass(SSSPVertex.class);
job.setCombinerClass(MinLongCombiner.class);
job.set(START_VERTEX, args[0]);
job.addInput(TableInfo.builder().tableName(args[1]).build());
job.addOutput(TableInfo.builder().tableName(args[2]).build());
long startTime = System.currentTimeMillis(); job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}
}

```

1.9.6.2. PageRank

This topic provides a PageRank example.

PageRank is a classic algorithm used to rank web pages. The input of the algorithm is a digraph G . Each vertex represents a web page, and the edge between two vertices represents the link between two web pages. Basic principles of the algorithm lie in the following aspects:

- Initialization: A vertex value indicates the rank value of PageRank. Vertex values are of the DOUBLE type. During initialization, the values of all vertices are $1/\text{TotalNumVertices}$.
- Iteration formula: $\text{PageRank}(i) = 0.15/\text{TotalNumVertices} + 0.85 \times \text{sum}$. In this formula, sum indicates the sum of the $\text{PageRank}(j)/\text{out_degree}(j)$ values. j indicates all vertices that point to vertex i .

The PageRank algorithm is suitable for MaxCompute Graph programs. Each vertex j maintains its PageRank value and sends the $\text{PageRank}(j)/\text{out_degree}(j)$ value to its adjacent vertices (for voting) in each iteration. In the next iteration, each vertex recalculates the PageRank value by using the iteration formula.

Example:

```
import java.io.IOException;
```

```
import org.apache.log4j.Logger;
import com.aliyun.odps.io.WritableRecord;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.io.DoubleWritable;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.io.Writable;

public class PageRank {
private final static Logger LOG = Logger.getLogger(PageRank.class);
/**
 * Define PageRankVertex, where:
 * The vertex value indicates the current PageRank value of the vertex (web page).
 * The compute() method uses the following iteration formula to update the vertex value:  $PageRank(i) = 0.15/TotalNumVertices + 0.85 \times sum$ .
 * The cleanup() method writes the vertex value and its PageRank value to the result table.
 */
public static class PageRankVertex extends
Vertex<Text, DoubleWritable, NullWritable, DoubleWritable> {
@Override
public void compute(
ComputeContext<Text, DoubleWritable, NullWritable, DoubleWritable> context, Iterable<DoubleWritable> messages) throws IOException {
if (context.getSuperstep() == 0) {
setValue(new DoubleWritable(1.0 / context.getTotalNumVertices()));
} else if (context.getSuperstep() >= 1) { double sum = 0;
for (DoubleWritable msg : messages) { sum += msg.get();
}
DoubleWritable vertexValue = new DoubleWritable( (0.15f / context.getTotalNumVertices()) + 0.85f * sum);
setValue(vertexValue);
}
if (hasEdges()) {
context.sendMessageToNeighbors(this, new DoubleWritable(getValue()
.get() / getEdges().size()));
}
}
@Override
public void cleanup(
WorkerContext<Text, DoubleWritable, NullWritable, DoubleWritable> context) throws IOException {
context.write(getId(), getValue());
}
}
/** Define the PageRankVertexReader class, load a graph, and resolve each record in the table into a vertex. The first
column of the table is the source vertices and other columns are the destination vertices.**/
public static class PageRankVertexReader extends
GraphLoader<Text, DoubleWritable, NullWritable, DoubleWritable> {
@Override public void load(
LongWritable recordNum, WritableRecord record,
MutationContext<Text, DoubleWritable, NullWritable, DoubleWritable> context) throws IOException {
PageRankVertex vertex = new PageRankVertex();
vertex.setValue(new DoubleWritable(0));
vertex.setId((Text) record.get(0));
```

```

// ...
System.out.println(record.get(0));
for (int i = 1; i < record.size(); i++) {
    Writable edge = record.get(i);
    System.out.println(edge.toString());
    if (!(edge.equals(NullWritable.get()))) {
        vertex.addEdge(new Text(edge.toString()), NullWritable.get());
    }
}
LOG.info("vertex eds size: " + (vertex.hasEdges() ? vertex.getEdges().size() : 0));
context.addVertexRequest(vertex);
}
}
private static void printUsage() {
    System.out.println("Usage: <in> <out> [Max iterations (default 30)]");
    System.exit(-1);
}
public static void main(String[] args) throws IOException { if (args.length < 2)
    printUsage();
    GraphJob job = new GraphJob();
    // Define GraphJob, and specify the implementation method of Vertex/GraphLoader, the maximum number of iterations (> 30 by default), and the input and output tables.
    job.setGraphLoaderClass(PageRankVertexReader.class);
    job.setVertexClass(PageRankVertex.class);
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addOutput(TableInfo.builder().tableName(args[1]).build());
    // default max iteration is 30
    job.setMaxIteration(30); if (args.length >= 3)
    job.setMaxIteration(Integer.parseInt(args[2]));
    long startTime = System.currentTimeMillis(); job.run();
    System.out.println("Job Finished in "
        + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}
}

```

1.9.6.3. K-means clustering

This topic provides a k-means clustering example.

K-means clustering is a basic clustering algorithm that is widely used. How k-means clustering works: Clustering is performed around k points in space, and the closest vertices are classified. The values of the clustering centers are updated in sequence by using iterations until the optimal clustering result is obtained.

Procedure to divide the sample set into k classes:

1. Select the initial centers of k classes.
2. In the i th iteration, select a sample, calculate its distance to k centers, and then classify the sample into the class of the center with the shortest distance.
3. Use the mean method to update the center value of the class.
4. For all the k centers, if the value remains unchanged or is less than a threshold after the update, the iteration ends. Otherwise, the iteration continues.

Example:

```

import java.io.DataInput; import java.io.DataOutput;
import java.io.IOException;
import org.apache.log4j.Logger;

```

```
import com.aliyun.odps.io.WritableRecord;
import com.aliyun.odps.graph.Aggregator;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.io.DoubleWritable;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.io.Tuple;
import com.aliyun.odps.io.Writable;
public class Kmeans {
private final static Logger LOG = Logger.getLogger(Kmeans.class);
/** Define the KmeansVertex class. The compute() method is simple. It calls the aggregate() method of the context object and passes in the value of the current vertex. The value is of the TUPLE type and expressed by vector. */
public static class KmeansVertex extends
Vertex<Text, Tuple, NullWritable, NullWritable> {
@Override
public void compute(
ComputeContext<Text, Tuple, NullWritable, NullWritable> context, Iterable<NullWritable> messages) throws IOException { context.aggregate(getValue());
}
}
/** Define the KmeansVertexReader class, load a graph, and parse each record in the table as a vertex. The transmitted value of recordNum is used as the vertex ID. The vertex value is a tuple that consists of all columns in the record. */
public static class KmeansVertexReader extends
GraphLoader<Text, Tuple, NullWritable, NullWritable> {
@Override
public void load(LongWritable recordNum, WritableRecord record, MutationContext<Text, Tuple, NullWritable, NullWritable> context) throws IOException {
KmeansVertex vertex = new KmeansVertex();
vertex.setId(new Text(String.valueOf(recordNum.get())));
vertex.setValue(new Tuple(record.getAll()));
context.addVertexRequest(vertex);
}
}
public static class KmeansAggrValue implements Writable {
Tuple centers = new Tuple();
Tuple sums = new Tuple();
Tuple counts = new Tuple();
@Override
public void write(DataOutput out) throws IOException {
centers.write(out);
sums.write(out); counts.write(out);
}
@Override
public void readFields(DataInput in) throws IOException {
centers = new Tuple();
centers.readFields(in);
sums = new Tuple();
sums.readFields(in);
counts = new Tuple();
counts.readFields(in);
}
```

```

}
@Override
public String toString() {
return "centers " + centers.toString() + ", sums " + sums.toString()
+ ", counts " + counts.toString();
}
}
/**
 * Define the KmeansAggregator class. This class encapsulates the main logic of the k-means clustering algorithm.
 * createInitialValue is the initial value (the center point for each of the k classes) that is created for each iteration. In
 the first iteration (superstep 0), the value of this parameter is the initial center point. In other iterations, the value i
 s the new center point when the previous iteration ends.
 * The aggregate() method calculates the distance from each vertex to the centers of different classes, classifies the
 vertex into the class of the nearest center, and updates sum and count of the class.
 * The merge() method combines sums and counts collected by each worker.
 * The terminate() method calculates a new center point based on sum and count of each class. If the distance betwe
 en the original and new center points is less than a threshold or the number of iterations reaches the upper limit, t
 he iteration ends, and False is returned. The final center point is written to the result table.
 **/
public static class KmeansAggregator extends Aggregator<KmeansAggrValue> {
@SuppressWarnings("rawtypes")
@Override
public KmeansAggrValue createInitialValue(WorkerContext context)
throws IOException {
KmeansAggrValue aggrVal = null;
if (context.getSuperstep() == 0) {
aggrVal = new KmeansAggrValue();
aggrVal.centers = new Tuple();
aggrVal.sums = new Tuple();
aggrVal.counts = new Tuple();
byte[] centers = context.readCacheFile("centers");
String lines[] = new String(centers).split("\n");
for (int i = 0;
i < lines.length; i++) { String[] ss = lines[i].split(",");
Tuple center = new Tuple();
Tuple sum = new Tuple();
for (int j = 0; j < ss.length; ++j) {
center.append(new DoubleWritable(Double.valueOf(ss[j].trim())));
sum.append(new DoubleWritable(0.0));
}
LongWritable count = new LongWritable(0);
aggrVal.sums.append(sum); aggrVal.counts.append(count);
aggrVal.centers.append(center);
}
} else {
aggrVal = (KmeansAggrValue) context.getLastAggregatedValue(0);
}
return aggrVal;
}
@Override
public void aggregate(KmeansAggrValue value, Object item) {
int min = 0;
double mindist = Double.MAX_VALUE;
Tuple point = (Tuple) item;
for (int i = 0;
i < value.centers.size();
i++) { Tuple center = (Tuple) value.centers.get(i);
//use Euclidean Distance to calculate cent

```

```
// use Euclidean distance, no need to calculate sqrt
double dist = 0.0d;
for (int j = 0; j < center.size(); j++) {
    double v = ((DoubleWritable) point.get(j)).get()
    - ((DoubleWritable) center.get(j)).get();
    dist += v * v;
}
if (dist < mindist) { mindist = dist; min = i;
}
}
// update sum and count
Tuple sum = (Tuple) value.sums.get(min);
for (int i = 0;
i < point.size(); i++) {
    DoubleWritable s = (DoubleWritable) sum.get(i); s.set(s.get() + ((DoubleWritable) point.get(i)).get());
}
LongWritable count = (LongWritable) value.counts.get(min);
count.set(count.get() + 1);
}
@Override
public void merge(KmeansAggrValue value, KmeansAggrValue partial) {
    for (int i = 0; i < value.sums.size(); i++) {
        Tuple sum = (Tuple) value.sums.get(i);
        Tuple that = (Tuple) partial.sums.get(i);
        for (int j = 0; j < sum.size(); j++) {
            DoubleWritable s = (DoubleWritable) sum.get(j);
            s.set(s.get() + ((DoubleWritable) that.get(j)).get());
        }
    }
    for (int i = 0; i < value.counts.size(); i++) {
        LongWritable count = (LongWritable) value.counts.get(i);
        count.set(count.get() + ((LongWritable) partial.counts.get(i)).get());
    }
}
@SuppressWarnings("rawtypes")
@Override
public boolean terminate(WorkerContext context, KmeansAggrValue value) throws IOException {
    // compute new centers
    Tuple newCenters = new Tuple(value.sums.size());
    for (int i = 0; i < value.sums.size(); i++) {
        Tuple sum = (Tuple) value.sums.get(i);
        Tuple newCenter = new Tuple(sum.size());
        LongWritable c = (LongWritable) value.counts.get(i);
        for (int j = 0; j < sum.size(); j++) {
            DoubleWritable s = (DoubleWritable) sum.get(j);
            double val = s.get() / c.get();
            newCenter.set(j, new DoubleWritable(val));
        }
        // reset sum for next iteration
        s.set(0.0d);
    }
    // reset count for next iteration
    c.set(0);
    newCenters.set(i, newCenter);
}
// update centers
Tuple oldCenters = value.centers; value.centers = newCenters;
LOG.info("old centers: " + oldCenters + ", new centers: " + newCenters);
// compare new/old centers
```

```

boolean converged = true;
for (int i = 0; i < value.centers.size() && converged; i++) {
    Tuple oldCenter = (Tuple) oldCenters.get(i);
    Tuple newCenter = (Tuple) newCenters.get(i); double sum = 0.0d;
    for (int j = 0; j < newCenter.size(); j++) {
        double v = ((DoubleWritable) newCenter.get(j)).get() - ((DoubleWritable) oldCenter.get(j)).get();
        sum += v * v;
    }
    double dist = Math.sqrt(sum);
    LOG.info("old center: " + oldCenter + ", new center: " + newCenter + ", dist: " + dist);
    // converge threshold for each center: 0.05
    converged = dist < 0.05d;
}
if (converged || context.getSuperstep() == context.getMaxIteration() - 1) {
    // converged or reach max iteration, output centers
    for (int i = 0; i < value.centers.size(); i++) { context.write(((Tuple) value.centers.get(i)).toArray()); }
}
// true means to terminate iteration
return true;
}
// false means to continue iteration
return false;
}
}
private static void printUsage() {
    System.out.println("Usage: <in> <out> [Max iterations (default 30)]");
    System.exit(-1);
}
/** Define GraphJob, and specify the implementation method of Vertex/GraphLoader/Aggregator, the maximum number of iterations (> 30 by default), and the input and output tables. */
public static void main(String[] args) throws IOException {
    if (args.length < 2)
        printUsage();
    GraphJob job = new GraphJob();
    job.setGraphLoaderClass(KmeansVertexReader.class);
    job.setRuntimePartitioning(false);
    // Define job.setRuntimePartitioning(false). For the k-means clustering algorithm, vertices do not need to be distributed for graph loading. RuntimePartitioning is set to False to improve the performance of graph loading.
    job.setVertexClass(KmeansVertex.class);
    job.setAggregatorClass(KmeansAggregator.class);
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addOutput(TableInfo.builder().tableName(args[1]).build());
    // default max iteration is 30
    job.setMaxIteration(30); if (args.length >= 3)
    job.setMaxIteration(Integer.parseInt(args[2]));
    long start = System.currentTimeMillis(); job.run();
    System.out.println("Job Finished in " + (System.currentTimeMillis() - start) / 1000.0 + " seconds");
}
}

```

1.9.6.4. BiPartiteMatching

This topic provides a BiPartiteMatching example.

A bipartite graph is a graph where vertices are divided into two sets and each edge connects a vertex in one set to a vertex in the other set. In a bipartite graph, a matching is a set of pairwise non-adjacent edges in which no two edges share a common vertex. Bipartite matching is often used for information matching in scenarios with clear supply and demand relationships, such as online dating websites.

Implementation process:

1. From the first vertex on the left, select another unmatched vertex by following an alternating path to find an augmenting path.
2. If the unmatched vertex is found, an augmenting path is found.
3. Update path information, increase the number of matched edges by 1, and stop searching.
4. If no augmenting path is found, begin with another vertex.

Example:

```
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import java.util.Random;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.io.Text;
import com.aliyun.odps.io.Writable;
import com.aliyun.odps.io.WritableRecord;
public class BipartiteMatching {
    private static final Text UNMATCHED = new Text("UNMATCHED");
    public static class TextPair implements Writable {
        public Text first; public Text second;
        public TextPair() { first = new Text();
            second = new Text();
        }
        public TextPair(Text first, Text second) {
            this.first = new Text(first);
            this.second = new Text(second);
        }
        @Override
        public void write(DataOutput out) throws IOException {
            first.write(out);
            second.write(out);
        }
        @Override
        public void readFields(DataInput in) throws IOException {
            first = new Text();
            first.readFields(in);
            second = new Text();
            second.readFields(in);
        }
        @Override
        public String toString() { return first + ": " + second;
        }
    }
}
```

```

}
public static class BipartiteMatchingVertexReader extends
GraphLoader<Text, TextPair, NullWritable, Text> {
@Override
public void load(LongWritable recordNum, WritableRecord record, MutationContext<Text, TextPair, NullWritable, Text> context) throws IOException {
BipartiteMatchingVertex vertex = new BipartiteMatchingVertex();
vertex.setId((Text) record.get(0));
vertex.setValue(new TextPair(UNMATCHED, (Text) record.get(1)));
String[] adjs = record.get(2).toString().split(",");
for (String adj:adjs) {
vertex.addEdge(new Text(adj), null);
}
context.addVertexRequest(vertex);
}
}
public static class BipartiteMatchingVertex extends
Vertex<Text, TextPair, NullWritable, Text> {
private static final Text LEFT = new Text("LEFT");
private static final Text RIGHT = new Text("RIGHT");
private static Random rand = new Random();
@Override
public void compute(
ComputeContext<Text, TextPair, NullWritable, Text> context, Iterable<Text> messages) throws IOException {
if (isMatched()) { voteToHalt();
return;
}
switch ((int) context.getSuperstep() % 4) {
case 0:
if (isLeft()) {
context.sendMessageToNeighbors(this, getId());
}
break;
case 1:
if (isRight()) {
Text luckyLeft = null;
for (Text message : messages) { if (luckyLeft == null) {
luckyLeft = new Text(message);
} else {
if (rand.nextInt(1) == 0) { luckyLeft.set(message);
}
}
}
if (luckyLeft != null) { context.sendMessage(luckyLeft, getId());
}
}
break;
case 2:
if (isLeft()) {
Text luckyRight = null;
for (Text msg : messages) { if (luckyRight == null) {
luckyRight = new Text(msg);
} else {
if (rand.nextInt(1) == 0) { luckyRight.set(msg);
}
}
}
if (luckyRight != null) {

```

```
setMatchVertex(luckyRight);
context.sendMessage(luckyRight, getId());
}
}
break; case 3:
if (isRight()) {
for (Text msg : messages) { setMatchVertex(msg);
}
}
break;
}
}
@Override
public void cleanup(
WorkerContext<Text, TextPair, NullWritable, Text> context) throws IOException {
context.write(getId(), getValue().first);
}
private boolean isMatched() {
return ! getValue().first.equals(UNMATCHED);
}
private boolean isLeft() {
return getValue().second.equals(LEFT);
}
private boolean isRight() {
return getValue().second.equals(RIGHT);
}
private void setMatchVertex(Text matchVertex) { getValue().first.set(matchVertex);
}
}
private static void printUsage() {
System.err.println("BipartiteMatching <input> <output> [maxIteration]");
}
public static void main(String[] args) throws IOException { if (args.length < 2) {
printUsage();
}
GraphJob job = new GraphJob();
job.setGraphLoaderClass(BipartiteMatchingVertexReader.class);
job.setVertexClass(BipartiteMatchingVertex.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
int maxIteration = 30;
if (args.length > 2) {
maxIteration = Integer.parseInt(args[2]);
}
job.setMaxIteration(maxIteration);
job.run();
}
}
```

1.9.6.5. Strongly connected components

This section provides an example of strongly connected components.

A directed graph is called a strongly connected graph if every vertex is reachable from every other vertex. A strongly connected sub-graph with a large number of vertices in a directed graph is called a strongly connected component.

- * 2. Trimming: Takes one superstep. Every vertex with only in-coming or only outgoing edges (or neither) sets its colorID to its own ID and becomes inactive. Messages subsequently sent to the vertex > are ignored.
- *
- * 3. Forward-Traversal: There are two sub phases: Start and Rest. In the Start phase, each vertex sets its colorID to its own ID and > propagates its ID to its outgoing neighbors. In the Rest phase, vertices update their own colorIDs with the minimum colorID they have seen, and > propagate their colorIDs, if updated, until the colorIDs converge.
- * Set the phase to Backward-Traversal when the colorIDs converge.
- *
- * 4. Backward-Traversal: We again break the phase into Start and Rest. In Start, every vertex whose ID equals its colorID propagates its ID > to the vertices in transposeNeighbors and sets itself inactive. > Messages subsequently sent to the vertex are ignored. In each of the Rest > phase supersteps, each vertex receiving a message that matches its colorID: (1) > propagates its colorID in the transpose graph; (2) sets itself inactive. > Messages subsequently sent to the vertex are ignored. Set the phase back to Trimming if not all vertex are inactive.
- *

* <http://ilpubs.stanford.edu:8090/1077/3/p535-salihoglu.pdf>

**/

```
public class StronglyConnectedComponents {
    public final static int STAGE_TRANSPOSE_1 = 0;
    public final static int STAGE_TRANSPOSE_2 = 1;
    public final static int STAGE_TRIMMING = 2;
    public final static int STAGE_FW_START = 3;
    public final static int STAGE_FW_REST = 4;
    public final static int STAGE_BW_START = 5;
    public final static int STAGE_BW_REST = 6;
    /**
```

* The value is composed of component id, incoming neighbors,
* active status and updated status.

**/

```
public static class MyValue implements Writable {
    LongWritable sccID;// strongly connected component id
    Tuple inNeighbors;// transpose neighbors
    BooleanWritable active;// vertex is active or not
    BooleanWritable updated;// sccID is updated or not
    public MyValue() {
        this.sccID = new LongWritable(Long.MAX_VALUE);
        this.inNeighbors = new Tuple();
        this.active = new BooleanWritable(true);
        this.updated = new BooleanWritable(false);
    }
    public void setSccID(LongWritable sccID) {
        this.sccID = sccID;
    }
    public LongWritable getSccID() {
        return this.sccID;
    }
    public void setInNeighbors(Tuple inNeighbors) {
        this.inNeighbors = inNeighbors;
    }
    public Tuple getInNeighbors() {
        return this.inNeighbors;
    }
    public void addInNeighbor(LongWritable neighbor) {
```

```

this.inNeighbors.append(new LongWritable(neighbor.get()));
}
public boolean isActive() {
return this.active.get();
}
public void setActive(boolean status) {
this.active.set(status);
}
public boolean isUpdated() {
return this.updated.get();
}
public void setUpdated(boolean update) {
this.updated.set(update);
}
@Override
public void write(DataOutput out) throws IOException {
this.sccID.write(out);
this.inNeighbors.write(out);
this.active.write(out);
this.updated.write(out);
}
@Override
public void readFields(DataInput in) throws IOException {
this.sccID.readFields(in);
this.inNeighbors.readFields(in);
this.active.readFields(in);
this.updated.readFields(in);
}
@Override
public String toString() {
StringBuilder sb = new StringBuilder();
sb.append("sccID: " + sccID.get());
sb.append(" inNeighbors: " + inNeighbors.toDelimitedString(','));
sb.append(" active: " + active.get());
sb.append(" updated: " + updated.get());
return sb.toString();
}
}
public static class SCCVertex extends
Vertex<LongWritable, MyValue, NullWritable, LongWritable> {
public SCCVertex() {
this.setValue(new MyValue());
}
@Override
public void compute(
ComputeContext<LongWritable, MyValue, NullWritable, LongWritable> context, Iterable<LongWritable> msgs) thro
ws IOException {
// Messages sent to inactive vertex are ignored.
if (! this.getValue().isActive()) {
this.voteToHalt(); return;
}
int stage = ((SCCAggrValue)context.getLastAggregatedValue(0)).getStage(); switch (stage) {
case STAGE_TRANSPOSE_1:
context.sendMessageToNeighbors(this, this.getId());
break;
case STAGE_TRANSPOSE_2:
for (LongWritable msg: msgs) {

```

```
this.getValue().addInNeighbor(msg);
}
case STAGE_TRIMMING:
this.getValue().setScclD(getId());
if (this.getValue().getInNeighbors().size() == 0 || this.getNumEdges() == 0) {
this.getValue().setActive(false);
}
break;
case STAGE_FW_START: this.getValue().setScclD(getId());
context.sendMessageToNeighbors(this, this.getValue().getScclD());
break;
case STAGE_FW_REST:
long minScclD = Long.MAX_VALUE;
for (LongWritable msg : msgs) {
if (msg.get() < minScclD) { minScclD = msg.get();
}
}
if (minScclD < this.getValue().getScclD().get()) {
this.getValue().setScclD(new LongWritable(minScclD));
context.sendMessageToNeighbors(this, this.getValue().getScclD());
this.getValue().setUpdated(true);
} else {
this.getValue().setUpdated(false);
}
break;
case STAGE_BW_START:
if (this.getId().equals(this.getValue().getScclD())) {
for (Writable neighbor : this.getValue().getInNeighbors().getAll()) {
context.sendMessage((LongWritable)neighbor, this.getValue().getScclD());
}
this.getValue().setActive(false);
}
break;
case STAGE_BW_REST: this.getValue().setUpdated(false);
for (LongWritable msg : msgs) {
if (msg.equals(this.getValue().getScclD())) {
for (Writable neighbor : this.getValue().getInNeighbors().getAll()) {
context.sendMessage((LongWritable)neighbor, this.getValue().getScclD());
}
this.getValue().setActive(false);
this.getValue().setUpdated(true);
break;
}
}
break;
}
context.aggregate(0, getValue());
}
@Override
public void cleanup(
WorkerContext<LongWritable, MyValue, NullWritable, LongWritable> context)
throws IOException {
context.write(getId(), getValue().getScclD());
}
}
/**
* The SCCAggrValue maintains global stage and graph updated and > active status.
* updated is true only if one vertex is updated.
```

```

* active is true only if one vertex is active.
*/
public static class SCCAggrValue implements Writable {
    IntWritable stage = new IntWritable(STAGE_TRANSPOSE_1);
    BooleanWritable updated = new BooleanWritable(false);
    BooleanWritable active = new BooleanWritable(false);
    public void setStage(int stage) { this.stage.set(stage);
    }
    public int getStage() { return this.stage.get();
    }
    public void setUpdated(boolean updated) {
    this.updated.set(updated);
    }
    public boolean getUpdated() {
    return this.updated.get();
    }
    public void setActive(boolean active) {
    this.active.set(active);
    }
    public boolean getActive() {
    return this.active.get();
    }
    @Override
    public void write(DataOutput out) throws IOException {
    this.stage.write(out);
    this.updated.write(out);
    this.active.write(out);
    }
    @Override
    public void readFields(DataInput in) throws IOException {
    this.stage.readFields(in);
    this.updated.readFields(in);
    this.active.readFields(in);
    }
}
/**
* The job of SCCAggregator is to schedule global stage in > every superstep.
*/
public static class SCCAggregator extends Aggregator<SCCAggrValue> {
    @SuppressWarnings("rawtypes")
    @Override
    public SCCAggrValue createStartupValue(WorkerContext context) throws IOException { return new SCCAggrValue();
    }
    @SuppressWarnings("rawtypes")
    @Override
    public SCCAggrValue createInitialValue(WorkerContext context) throws IOException {
    return (SCCAggrValue) context.getLastAggregatedValue(0);
    }
    @Override
    public void aggregate(SCCAggrValue value, Object item) throws IOException { MyValue v = (MyValue)item;
    if ((value.getStage() == STAGE_FW_REST || value.getStage() == STAGE_BW_REST)&& v.isUpdated()) { value.setUpdated(true);
    }
    // only active vertex invoke aggregate()
    value.setActive(true);
    }
    @Override
    public void reduce(SCCAggrValue value, SCCAggrValue initial) throws IOException {

```

```
public void merge(SCCAggrValue value, SCCAggrValue partial) throws IOException {
    boolean updated = value.getUpdated() || partial.getUpdated();
    value.setUpdated(updated);
    boolean active = value.getActive() || partial.getActive();
    value.setActive(active);
}
@SuppressWarnings("rawtypes")
@Override
public boolean terminate(WorkerContext context, SCCAggrValue value) throws IOException {
    // If all vertices is inactive, job is over.
    if (! value.getActive()) { return true;
    }
    // state machine
    switch (value.getStage()) {
    case STAGE_TRANSPOSE_1: value.setStage(STAGE_TRANSPOSE_2);
    break;
    case STAGE_TRANSPOSE_2: value.setStage(STAGE_TRIMMING);
    break;
    case STAGE_TRIMMING: value.setStage(STAGE_FW_START);
    break;
    case STAGE_FW_START: value.setStage(STAGE_FW_REST);
    break;
    case STAGE_FW_REST: if (value.getUpdated()) {
    value.setStage(STAGE_FW_REST);
    } else {
    value.setStage(STAGE_BW_START);
    }
    break;
    case STAGE_BW_START: value.setStage(STAGE_BW_REST);
    break;
    case STAGE_BW_REST: if (value.getUpdated()) { value.setStage(STAGE_BW_REST);
    } else { value.setStage(STAGE_TRIMMING);
    }
    break;
    }
    value.setActive(false);
    value.setUpdated(false);
    return false;
}
}
public static class SCCVertexReader extends
GraphLoader<LongWritable, MyValue, NullWritable, LongWritable> {
    @Override public void load(
    LongWritable recordNum, WritableRecord record,
    MutationContext<LongWritable, MyValue, NullWritable, LongWritable> context) throws IOException {
    SCCVertex vertex = new SCCVertex();
    vertex.setID((LongWritable) record.get(0));
    String[] edges = record.get(1).toString().split(",");
    for (int i = 0; i < edges.length; i++) { try {
    long destID = Long.parseLong(edges[i]);
    vertex.addEdge(new LongWritable(destID), NullWritable.get());
    } catch (NumberFormatException nfe) { System.err.println("Ignore " + nfe);
    }
    }
    context.addVertexRequest(vertex);
    }
}
public static void main(String[] args) throws IOException {
```

```

if (args.length < 2) {
    System.out.println("Usage: <input> <output>");
    System.exit(-1);
}
GraphJob job = new GraphJob();
job.setGraphLoaderClass(SCCVertexReader.class);
job.setVertexClass(SCCVertex.class);
job.setAggregatorClass(SCCAggregator.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
long startTime = System.currentTimeMillis();
job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}

```

1.9.6.6. Connected components

This section provides an example of connected components.

Two vertices are connected if a path exists between them. If each vertex in an undirected graph G is connected to all the other vertices in the graph, G is called a connected graph. Otherwise, G is called an unconnected graph. A connected sub-graph with a large number of vertices is called a connected component.

This algorithm calculates connected component members of each vertex, and generates the connected component of the vertex value that includes the smallest vertex ID. The smallest vertex ID is propagated along edges to all vertices of the connected component.

Example:

```

import java.io.IOException;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.graph.examples.SSSP.MinLongCombiner;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.io.WritableRecord;
/**
 * Compute the connected component membership of each vertex and output
 * each vertex which's value containing the smallest id in the > connected
 * component containing that vertex.
 *
 * Algorithm: propagate the smallest vertex id along the edges to all
 * vertices of a connected component.
 */
public class ConnectedComponents {
    public static class CCVertex extends
        Vertex<LongWritable, LongWritable, NullWritable, LongWritable> {
        @Override
        public void compute(

```

```

ComputeContext<LongWritable, LongWritable, NullWritable, LongWritable> context, Iterable<LongWritable> msgs)
throws IOException {
    if (context.getSuperstep() == 0L) {
        this.setValue(getId());
        context.sendMessageToNeighbors(this, getValue());
        return;
    }
    long minID = Long.MAX_VALUE;
    for (LongWritable id : msgs) {
        if (id.get() < minID) { minID = id.get(); }
    }
    if (minID < this.getValue().get()) {
        this.setValue(new LongWritable(minID));
        context.sendMessageToNeighbors(this, getValue());
    } else {
        this.voteToHalt();
    }
}
/**
 * Output Table Description:
 * +-----+-----+
 * Field | Type | Comment |
 * +-----+-----+
 * v | bigint | vertex id |
 * minID | bigint | smallest id in the connected component |
 * +-----+-----+
 */
@Override
public void cleanup(
    WorkerContext<LongWritable, LongWritable, NullWritable, LongWritable> context) throws IOException {
    context.write(getId(), getValue());
}
/**
 * Input Table Description:
 * +-----+-----+
 * Field | Type | Comment |
 * +-----+-----+
 * v | bigint | vertex id |
 * es | string | comma separated target vertex id of outgoing edges |
 * +-----+-----+
 *
 * Example:
 * For graph:
 * 1 ---- 2
 * |
 * 3 ---- 4
 * Input table:
 * +-----+
 * v | es |
 * +-----+
 * 1 | 2,3 |
 * 2 | 1,4 |
 * 3 | 1,4 |
 * 4 | 2,3 |
 * +-----+
 */

```

```

public static class CCVertexReader extends
GraphLoader<LongWritable, LongWritable, NullWritable, LongWritable> {
@Override
public void load(
LongWritable recordNum, WritableRecord record,
MutationContext<LongWritable, LongWritable, NullWritable, LongWritable> context) throws IOException {
CCVertex vertex = new CCVertex();
vertex.setId((LongWritable) record.get(0));
String[] edges = record.get(1).toString().split(",");
for (int i = 0; i < edges.length; i++) {
long destID = Long.parseLong(edges[i]);
vertex.addEdge(new LongWritable(destID), NullWritable.get());
}
context.addVertexRequest(vertex);
}
}
public static void main(String[] args) throws IOException {
if (args.length < 2) {
System.out.println("Usage: <input> <output>");
System.exit(-1);
}
GraphJob job = new GraphJob();
job.setGraphLoaderClass(CCVertexReader.class);
job.setVertexClass(CCVertex.class);
job.setCombinerClass(MinLongCombiner.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
long startTime = System.currentTimeMillis();
job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}

```

1.9.6.7. Topological sorting

This topic provides an example of topological sorting in MaxCompute Graph.

For a directed edge (u,v) , all vertex sequences that satisfy $u < v$ are called topological sequences. Topological sorting is an algorithm that is used to calculate the topological sequence of a directed graph.

Steps to implement the algorithm:

1. Identify a vertex without incoming edges and generate an output record.
2. Delete the vertex and all its outgoing edges from the graph.
3. Repeat the preceding steps until output records are generated for all the vertices without incoming edges.

Example:

```

import java.io.IOException;
import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.Aggregator;
import com.aliyun.odps.graph.Combiner;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;

```

```
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.io.BooleanWritable;
import com.aliyun.odps.io.WritableRecord;
public class TopologySort {
private final static Log LOG = LogFactory.getLog(TopologySort.class);
public static class TopologySortVertex extends
Vertex<LongWritable, LongWritable, NullWritable, LongWritable> {
@Override
public void compute(
ComputeContext<LongWritable, LongWritable, NullWritable, LongWritable> context, Iterable<LongWritable> messages) throws IOException {
// in superstep 0, each vertex sends message whose value is 1 to its
// neighbors
if (context.getSuperstep() == 0) { if (hasEdges()) {
context.sendMessageToNeighbors(this, new LongWritable(1L));
}
} else if (context.getSuperstep() >= 1) {
// compute each vertex's indegree
long indegree = getValue().get();
for (LongWritable msg : messages) {
indegree += msg.get();
}
setValue(new LongWritable(indegree));
if (indegree == 0) {
voteToHalt();
if (hasEdges()) {
context.sendMessageToNeighbors(this, new LongWritable(-1L));
}
context.write(new LongWritable(context.getSuperstep()), getId());
LOG.info("vertex: " + getId());
}
context.aggregate(new LongWritable(indegree));
}
}
}
public static class TopologySortVertexReader extends
GraphLoader<LongWritable, LongWritable, NullWritable, LongWritable> {
@Override public void load(
LongWritable recordNum, WritableRecord record,
MutationContext<LongWritable, LongWritable, NullWritable, LongWritable> context) throws IOException {
TopologySortVertex vertex = new TopologySortVertex();
vertex.setId((LongWritable) record.get(0));
vertex.setValue(new LongWritable(0));
String[] edges = record.get(1).toString().split(",");
for (int i = 0; i < edges.length; i++) {
long edge = Long.parseLong(edges[i]);
if (edge >= 0) {
vertex.addEdge(new LongWritable(Long.parseLong(edges[i])), NullWritable.get());
}
}
LOG.info(record.toString());
context.addVertexRequest(vertex);
}
}
```

```

}
}
public static class LongSumCombiner extends
Combiner<LongWritable, LongWritable> {
@Override
public void combine(LongWritable vertexId, LongWritable combinedMessage, LongWritable messageToCombine)
throws IOException {
combinedMessage.set(combinedMessage.get() + messageToCombine.get());
}
}
public static class TopologySortAggregator extends
Aggregator<BooleanWritable> {
@SuppressWarnings("rawtypes")
@Override
public BooleanWritable createInitialValue(WorkerContext context) throws IOException {
return new BooleanWritable(true);
}
@Override
public void aggregate(BooleanWritable value, Object item) throws IOException {
boolean hasCycle = value.get();
boolean inDegreeNotZero = ((LongWritable) item).get() == 0 ? false : true;
value.set(hasCycle && inDegreeNotZero);
}
@Override
public void merge(BooleanWritable value, BooleanWritable partial) throws IOException {
value.set(value.get() && partial.get());
}
@SuppressWarnings("rawtypes")
@Override
public boolean terminate(WorkerContext context, BooleanWritable value) throws IOException {
if (context.getSuperstep() == 0) {
// since the initial aggregator value is true, and in superstep we don't
// do aggregate
return false;
}
return value.get();
}
}
public static void main(String[] args) throws IOException { if (args.length != 2) {
System.out.println("Usage : <inputTable> <outputTable>");
System.exit(-1);
}
// Format of the input table:
// 0 1, 2
// 1 3
// 2 3
// 3 -1
// The first column is vertexid, and the second column is the destination vertexid of the vertex. If the value is -1, the
vertex does not have any outgoing edges.
// Format of the output table:
// 0 0
// 1 1
// 1 2
// 2 3
// The first column is the supstep value, in which the topological sequence is hidden. The second column is vertexid.
// TopologySortAggregator is used to determine whether the graph has loops.
// If the input graph has a loop and the indegree of all active vertices is not 0, the iteration ends.
// You can use records in the input and output tables to determine whether the graph has loops.

```

```
GraphJob job = new GraphJob();
job.setGraphLoaderClass(TopologySortVertexReader.class);
job.setVertexClass(TopologySortVertex.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
job.setCombinerClass(LongSumCombiner.class);
job.setAggregatorClass(TopologySortAggregator.class);
long startTime = System.currentTimeMillis(); job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}
```

1.9.6.8. Linear regression

This topic provides a linear regression example.

In statistics, linear regression is a statistical analysis method used to determine the dependency between two or more variables. Linear regression is different from the classification algorithm that predicts discrete values. The regression algorithm can predict continuous values.

The linear regression algorithm defines the loss function as the sum of the least square errors of a sample set. It solves the weight vector by minimizing the loss function.

A common solution is the gradient descent method. It is implemented in the following steps:

1. Initialize the weight vector to provide the descent speed and iterations or iteration convergence condition.
2. Calculate the least square error for each sample.
3. Calculate the sum of the least square errors and update the weight based on the descent speed.
4. Repeat iterations until convergence occurs.

Example:

```
import java.io.DataInput;
import java.io.DataOutput;
import java.io.IOException;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.Aggregator;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.io.DoubleWritable;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.io.Tuple;
import com.aliyun.odps.io.Writable;
import com.aliyun.odps.io.WritableRecord;
/**
 * LineRegression input: y,x1,x2,x3,.....
 *
 * @author shiwan.ch
 * @update jiasheng.tjs running parameters are like: tjs_lr_in > tjs_lr_out 1500 2
 * 0.07
 */
```

```

public class LinearRegression {
    public static class GradientWritable implements Writable {
        Tuple lastTheta;
        Tuple currentTheta;
        Tuple tmpGradient;
        LongWritable count;
        DoubleWritable lost;
        @Override
        public void readFields(DataInput in) throws IOException {
            lastTheta = new Tuple();
            lastTheta.readFields(in);
            currentTheta = new Tuple();
            currentTheta.readFields(in);
            tmpGradient = new Tuple();
            tmpGradient.readFields(in);
            count = new LongWritable();
            count.readFields(in);
            /* update 1: add a variable to store lost at every iteration */
            lost = new DoubleWritable();
            lost.readFields(in);
        }
        @Override
        public void write(DataOutput out) throws IOException {
            lastTheta.write(out);
            currentTheta.write(out);
            tmpGradient.write(out);
            count.write(out);
            lost.write(out);
        }
    }
    public static class LinearRegressionVertex extends
        Vertex<LongWritable, Tuple, NullWritable, NullWritable> {
        @Override
        public void compute(
            ComputeContext<LongWritable, Tuple, NullWritable, NullWritable> context, Iterable<NullWritable> messages) throws
            IOException {
            context.aggregate(getValue());
        }
    }
    public static class LinearRegressionVertexReader extends
        GraphLoader<LongWritable, Tuple, NullWritable, NullWritable> {
        @Override
        public void load(LongWritable recordNum, WritableRecord record, MutationContext<LongWritable, Tuple, NullWritable,
            NullWritable> context)
            throws IOException {
            LinearRegressionVertex vertex = new LinearRegressionVertex();
            vertex.setId(recordNum);
            vertex.setValue(new Tuple(record.getAll())); context.addVertexRequest(vertex);
        }
    }
    public static class LinearRegressionAggregator extends
        Aggregator<GradientWritable> {
        @SuppressWarnings("rawtypes")
        @Override
        public GradientWritable createInitialValue(WorkerContext context) throws IOException {
            if (context.getSuperstep() == 0) {
                /* set initial value, all 0 */
                GradientWritable grad = new GradientWritable();

```

```
GradientWritable grad = new GradientWritable();
grad.lastTheta = new Tuple();
grad.currentTheta = new Tuple();
grad.tmpGradient = new Tuple();
grad.count = new LongWritable(1);
grad.lost = new DoubleWritable(0.0);
int n = (int) Long.parseLong(context.getConfiguration().get("Dimension"));
for (int i = 0; i < n; i++) { grad.lastTheta.append(new DoubleWritable(0));
grad.currentTheta.append(new DoubleWritable(0));
grad.tmpGradient.append(new DoubleWritable(0));
}
return grad;
} else
return (GradientWritable) context.getLastAggregatedValue(0);
}
public static double vecMul(Tuple value, Tuple theta) {
/* perform this partial computing: y(i)-hθ(x(i)) for each sample */
/* value denote a piece of sample and value(0) is y */
double sum = 0.0;
for (int j = 1; j < value.size(); j++)
sum += Double.parseDouble(value.get(j).toString()) * Double.parseDouble(theta.get(j).toString());
Double tmp = Double.parseDouble(theta.get(0).toString()) + sum -Double.parseDouble(value.get(0).toString());
return tmp;
}
@Override
public void aggregate(GradientWritable gradient, Object value) throws IOException {
/*
* perform on each vertex--each sample i: set theta(j) for each sample > i
* for each dimension
*/
double tmpVar = vecMul((Tuple) value, gradient.currentTheta);
/*
* update 2:local worker aggregate(), perform like merge() below. This
* means the variable gradient denotes the previous aggregated value
*/
gradient.tmpGradient.set(0, new DoubleWritable( ((DoubleWritable) gradient.tmpGradient.get(0)).get() + tmpVar));
gradient.lost.set(Math.pow(tmpVar, 2));
/*
* calculate (y(i)-hθ(x(i)))x(i)(j) for each sample i for each
* dimension j
*/
for (int j = 1; j < gradient.tmpGradient.size(); j++) gradient.tmpGradient.set(j, new DoubleWritable(
(((DoubleWritable) gradient.tmpGradient.get(j)).get() + tmpVar * Double.parseDouble(((Tuple) value).get(j).toString
()))));
}
@Override
public void merge(GradientWritable gradient, GradientWritable partial) throws IOException {
/* perform SumAll on each dimension for all samples. */
Tuple master = (Tuple) gradient.tmpGradient;
Tuple part = (Tuple) partial.tmpGradient;
for (int j = 0; j < gradient.tmpGradient.size(); j++) {
DoubleWritable s = (DoubleWritable) master.get(j);
s.set(s.get() + ((DoubleWritable) part.get(j)).get());
}
gradient.lost.set(gradient.lost.get() + partial.lost.get());
}
@SuppressWarnings("rawtypes")
@Override
```

```

public boolean terminate(WorkerContext context, GradientWritable gradient) throws IOException {
    /*
    * 1. calculate new theta 2. judge the diff between last step and this
    * step, if smaller than the threshold, stop iteration
    */
    gradient.set(new DoubleWritable(gradient.get() / (2 * context.getTotalNumVertices())));
    /*
    * we can calculate lost in order to make sure the algorithm is running > on
    * the right direction (for debug)
    */
    System.out.println(gradient.count + " lost:" + gradient.get());
    Tuple tmpGradient = gradient.tmpGradient;
    System.out.println("tmpGra" + tmpGradient);
    Tuple lastTheta = gradient.lastTheta;
    Tuple tmpCurrentTheta = new Tuple(gradient.currentTheta.size());
    System.out.println(gradient.count + " terminate_start_last:" + lastTheta);
    double alpha = 0.07; // learning rate
    // alpha =
    // Double.parseDouble(context.getConfiguration().get("Alpha"));
    /* perform theta(j) = theta(j) - alpha * tmpGradient */
    long M = context.getTotalNumVertices();
    /*
    * update 3: add (/M) on the code. The original code forget this step
    */
    for (int j = 0; j < lastTheta.size(); j++) { tmpCurrentTheta
        .set(j,
        new DoubleWritable(Double.parseDouble(lastTheta.get(j)
        .toString()) - alpha / M * Double.parseDouble(tmpGradient.get(j).toString())));
    }
    System.out.println(gradient.count + " terminate_start_current:" + tmpCurrentTheta);
    // judge if convergence is happening.
    double diff = 0.00d;
    for (int j = 0; j < gradient.currentTheta.size(); j++)
        diff += Math.pow(((DoubleWritable) tmpCurrentTheta.get(j)).get() - ((DoubleWritable) lastTheta.get(j)).get(), 2);
    if (
    /*
    * Math.sqrt(diff) < 0.00000000005d ||
    */
    Long.parseLong(context.getConfiguration().get("Max_Iter_Num")) == gradient.count
    .get()) { context.write(gradient.currentTheta.toArray());
        return true;
    }
    gradient.lastTheta = tmpCurrentTheta;
    gradient.currentTheta = tmpCurrentTheta;
    gradient.count.set(gradient.count.get() + 1);
    int n = (int) Long.parseLong(context.getConfiguration().get("Dimension"));
    /*
    * update 4: Important!!! Remember this step. Graph won't reset the
    * initial value for global variables at the beginning of each iteration
    */
    for (int i = 0; i < n; i++) {
        gradient.tmpGradient.set(i, new DoubleWritable(0));
    }
    return false;
    }
    }
    public static void main(String[] args) throws IOException { GraphJob job = new GraphJob();
    job.setGraphLoaderClass(InearRegressionVertexReader.class); job.setRuntimePartitioning(false);
}

```

```
job.setGraphLoaderClass(LinearRegressionVertexLoader.class); job.setParameters(learningRate);
job.setNumWorkers(3);
job.setVertexClass(LinearRegressionVertex.class);
job.setAggregatorClass(LinearRegressionAggregator.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
job.setMaxIteration(Integer.parseInt(args[2])); // Numbers of Iteration
job.setInt("Max_Iter_Num", Integer.parseInt(args[2]));
job.setInt("Dimension", Integer.parseInt(args[3])); // Dimension
job.setFloat("Alpha", Float.parseFloat(args[4])); // Learning rate
long start = System.currentTimeMillis(); job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - start) / 1000.0 + " seconds");
}
}
```

1.9.6.9. Triangle count

This topic provides a triangle count example.

The triangle count algorithm calculates the number of triangles that pass through each vertex in a graph. The algorithm is implemented in the following steps:

1. Send the ID of each vertex to all its outgoing neighbors.
2. Store information about incoming and outgoing neighbors, and send the information to outgoing neighbors.
3. Calculate the number of endpoint intersections for each edge, calculate the sum, and write the output results to a table.
4. Sum up the output results in the table and divide the sum by 3 to obtain the number of triangles that pass through each vertex.

Example:

```
import java.io.IOException;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.Edge;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.WorkerContext;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.NullWritable;
import com.aliyun.odps.io.Tuple;
import com.aliyun.odps.io.Writable;
import com.aliyun.odps.io.WritableRecord;
/**
 * Compute the number of triangles passing through each vertex.
 *
 * The algorithm can be computed in three supersteps:
 * I. Each vertex sends a message with its ID to all its outgoing
 * neighbors.
 * II. The incoming neighbors and outgoing neighbors are stored and
 * send to outgoing neighbors.
 * III. For each edge compute the intersection of the sets at destination
 * vertex and sum them, then output to table.
 *
 */
```

```
* The triangle count is the sum of output table and divide by three > since
* each triangle is counted three times.
*
**/
public class TriangleCount {
    public static class TCVertex extends
    Vertex<LongWritable, Tuple, NullWritable, Tuple> {
        @Override
        public void setup(
        WorkerContext<LongWritable, Tuple, NullWritable, Tuple> context) throws IOException {
            // collect the outgoing neighbors
            Tuple t = new Tuple();
            if (this.hasEdges()) {
                for (Edge<LongWritable, NullWritable> edge : this.getEdges()) {
                    t.append(edge.getDestVertexId());
                }
            }
            this.setValue(t);
        }
        @Override
        public void compute(
        ComputeContext<LongWritable, Tuple, NullWritable, Tuple> context, Iterable<Tuple> msgs) throws IOException {
            if (context.getSuperstep() == 0L) {
                // sends a message with its ID to all its outgoing neighbors
                Tuple t = new Tuple(); t.append(getId());
                context.sendMessageToNeighbors(this, t);
            } else if (context.getSuperstep() == 1L) {
                // store the incoming neighbors
                for (Tuple msg : msgs) {
                    for (Writable item : msg.getAll()) {
                        if (! this.getValue().getAll().contains((LongWritable)item)) {
                            this.getValue().append((LongWritable)item);
                        }
                    }
                }
            }
            // send both incoming and outgoing neighbors to all outgoing neighbors
            context.sendMessageToNeighbors(this, getValue());
            } else if (context.getSuperstep() == 2L) {
                // count the sum of intersection at each edge
                long count = 0;
                for (Tuple msg : msgs) {
                    for (Writable id : msg.getAll()) {
                        if (getValue().getAll().contains(id)) { count ++; }
                    }
                }
            }
            // output to table
            context.write(getId(), new LongWritable(count));
            this.voteToHalt();
        }
    }
    public static class TCVertexReader extends
    GraphLoader<LongWritable, Tuple, NullWritable, Tuple> {
        @Override public void load(
        LongWritable recordNum, WritableRecord record,
        MutationContext<LongWritable, Tuple, NullWritable, Tuple> context) throws IOException {
            // ...
        }
    }
}
```

```
TCVertex vertex = new TCVertex();
vertex.setId((LongWritable) record.get(0));
String[] edges = record.get(1).toString().split(",");
for (int i = 0; i < edges.length; i++) { try {
    long destID = Long.parseLong(edges[i]);
    vertex.addEdge(new LongWritable(destID), NullWritable.get());
} catch (NumberFormatException nfe) { System.err.println("Ignore " + nfe);
}
}
context.addVertexRequest(vertex);
}
}
public static void main(String[] args) throws IOException { if (args.length < 2) {
    System.out.println("Usage: <input> <output>"); System.exit(-1);
}
GraphJob job = new GraphJob();
job.setGraphLoaderClass(TCVertexReader.class);
job.setVertexClass(TCVertex.class);
job.addInput(TableInfo.builder().tableName(args[0]).build());
job.addOutput(TableInfo.builder().tableName(args[1]).build());
long startTime = System.currentTimeMillis();
job.run();
System.out.println("Job Finished in " + (System.currentTimeMillis() - startTime) / 1000.0 + " seconds");
}
}
```

1.9.6.10. Edge table import

This topic provides an example of importing an edge table.

Example:

```
import java.io.IOException;
import com.aliyun.odps.conf.Configuration;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.graph.ComputeContext;
import com.aliyun.odps.graph.GraphJob;
import com.aliyun.odps.graph.GraphLoader;
import com.aliyun.odps.graph.Vertex;
import com.aliyun.odps.graph.VertexResolver;
import com.aliyun.odps.graph.MutationContext;
import com.aliyun.odps.graph.VertexChanges;
import com.aliyun.odps.graph.Edge;
import com.aliyun.odps.io.LongWritable;
import com.aliyun.odps.io.WritableComparable;
import com.aliyun.odps.io.WritableRecord;
/**
 * This example describes how to compile a Graph job program to load data of different data types. It covers how GraphLoader
 * and VertexResolver are used together to build a graph.
 *
 * A MaxCompute Graph job uses MaxCompute tables as the input. For example, a job uses two tables as the input.
 * One stores information about vertices, and the other stores information about edges.
 * Format of the table that stores information about vertices:
 * +-----+
 * | VertexID | VertexValue |
 * +-----+
```

```

* | id0| 9|
* +-----+
* | id1| 7|
* +-----+
* | id2| 8|
* +-----+
*
* Format of the table that stores information about edges:
* +-----+
* | VertexID | DestVertexID| EdgeValue|
* +-----+
* | id0| id1| 1|
* +-----+
* | id0| id2| 2|
* +-----+
* | id2| id1| 3|
* +-----+
*
* The two tables show that id0 has two outgoing edges that point to id1 and id2. id2 has one outgoing edge that points to id1, and id1 has no outgoing edges.
*
* For data of this type, in GraphLoader::load(LongWritable, Record, MutationContext), MutationContext#addVertexRequest(Vertex) can be used to add vertices to the graph.
* link MutationContext#addEdgeRequest(WritableComparable, Edge) can be used to add edges to the graph. In
* link VertexResolver#resolve(WritableComparable, Vertex, VertexChanges, boolean),
* vertices and edges added by using the load() method are combined to a vertex object. This object is used as the return value and added to the graph for computing.
*
**/
public class VertexInputFormat {
    private final static String EDGE_TABLE = "edge.table";
    /**
     * Resolve records to vertices and edges. Each record indicates a vertex or edge based on its source.
     * <p>
     * The following process is similar to the map stage of com.aliyun.odps.mapreduce.Mapper.
     * Enter a record to generate key-value pairs.
     * The keys are vertex IDs, and the values are vertices or edges that are written based on context. These key-value pairs are aggregated by using LoadingVertexResolver based on the vertex IDs.
     *
     * Note: The vertices or edges added here are requests sent based on the record content and are not used in computing.
     * Only the vertices or edges added by using VertexResolver are used for computing.
     */
    public static class VertexInputLoader extends
        GraphLoader<LongWritable, LongWritable, LongWritable, LongWritable> {
        private boolean isEdgeData;
        /**
         * Configure VertexInputLoader.
         *
         * @param conf
         * Indicates the configured parameters of a job. These parameters are configured in the main function of GraphJob or by running the SET command on the client.
         * @param workerId
         * Indicates the serial number of the running worker. The number starts from 0 and can be used to build a unique vertex ID.
         * @param inputTableInfo
         * Indicates the information about the input table that is loaded to the running worker. The information can be used to determine the data type of the input data (the format of the record)

```

```
e used to determine the data type of the input data (the format of the record).
**/
@Override
public void setup(Configuration conf, int workerId, TableInfo inputTableInfo) {
    isEdgeData = conf.get(EDGE_TABLE).equals(inputTableInfo.getTableName());
}
/**
 * Resolve the record to edges based on the record content and send a request to add them to the graph.
 *
 * @param recordNum
 *     Indicates the serial number of the record. The number starts from 1 and is separately counted in each work
er.
 * @param record
 *     Indicates the records in the input table. The table contains three columns, which indicate the source vertex,
destination vertex, and edge weight.
 * @param context
 *     Indicates the context. The context is used when you send a request to add resolved edges to the graph.
**/
@Override
public void load(
    LongWritable recordNum,
    WritableRecord record,
    MutationContext<LongWritable, LongWritable, LongWritable, LongWritable> context)
    throws IOException {
    if (isEdgeData) {
        /**
         * Data comes from the table that stores information about edges.
         *
         * 1. The first column indicates the IDs of source vertices.
         **/
        LongWritable sourceVertexID = (LongWritable) record.get(0);
        /**
         * 2. The second column indicates the IDs of destination vertices.
         **/
        LongWritable destinationVertexID = (LongWritable) record.get(1);
        /**
         * 3. The third column indicates edge weights.
         **/
        LongWritable edgeValue = (LongWritable) record.get(2);
        /**
         * 4. Create an edge based on a destination vertex ID and an edge weight.
         **/
        Edge<LongWritable, LongWritable> edge = new Edge<LongWritable, LongWritable>(
            destinationVertexID, edgeValue);
        /**
         * 5. Send a request to add the edge to a source vertex.
         **/
        context.addEdgeRequest(sourceVertexID, edge);
        /**
         * 6. If each record indicates a bidirectional edge, repeat Step 4 and Step 5.
         * Edge<LongWritable, LongWritable> edge2 = new
         * Edge<LongWritable, LongWritable>( sourceVertexID, edgeValue);
         * context.addEdgeRequest(destinationVertexID, edge2);
         **/
    } else {
        /**
         * Data comes from the table that stores information about vertices.
         *

```



```
not used in subsequent computing.
*
* @param vertexId
* Indicates the ID of the vertex that is requested to be added or the source vertex ID of the edge that is requested to be added.
* @param vertex
* Indicates an existing vertex object. The value of this parameter is always null in the data loading phase.
* @param vertexChanges
* Indicates a collection of vertices or edges that are requested to be added or removed for the ID.
* @param hasMessages
* Indicates whether messages are sent to the ID. The value of this parameter is always false in the data loading phase.
**/
@Override
public Vertex<LongWritable, LongWritable, LongWritable, LongWritable> resolve(
    LongWritable vertexId,
    Vertex<LongWritable, LongWritable, LongWritable, LongWritable> vertex,
    VertexChanges<LongWritable, LongWritable, LongWritable, LongWritable> vertexChanges,
    boolean hasMessages) throws IOException {
    /**
     * 1. Obtain the vertex object for computing.
     */
    MyVertex computeVertex = null;
    if (vertexChanges.getAddedVertexList() == null
        || vertexChanges.getAddedVertexList().isEmpty()) {
        computeVertex = new MyVertex();
        computeVertex.setId(vertexId);
    } else {
        /**
         * Each record indicates a unique vertex in the table that stores information about vertices.
         */
        computeVertex = (MyVertex) vertexChanges.getAddedVertexList().get(0);
    }
    /**
     * 2. Add the edge, which is requested to be added to the vertex, to the vertex object. If the data is a possible duplicate, deduplicate it based on algorithm requirements.
     */
    if (vertexChanges.getAddedEdgeList() != null) {
        for (Edge<LongWritable, LongWritable> edge : vertexChanges.getAddedEdgeList()) {
            computeVertex.addEdge(edge.getDestVertexId(), edge.getValue());
        }
    }
    /**
     * 3. Return the vertex object and add it to the final graph for computing.
     */
    return computeVertex;
}
}
/**
 * Determine the actions of the vertex that is used in computing.
 */
public static class MyVertex extends
    Vertex<LongWritable, LongWritable, LongWritable, LongWritable> {
    /**
     * Write the edge of the vertex to the result table based on the format of the input table. Make sure that the formats and data of the input and output tables are the same.
     */

```

```

*
* @param context
*   Indicates the runtime context.
* @param messages
*   Indicates the input messages.
**/
@Override
public void compute(
    ComputeContext<LongWritable, LongWritable, LongWritable, LongWritable> context,
    Iterable<LongWritable> messages) throws IOException {
    /**
     * Write the ID and value of the vertex to the result table that stores information about vertices.
     **/
    context.write("vertex", getId(), getValue());
    /**
     * Write the edge of the vertex to the result table that stores information about edges.
     **/
    if (hasEdges()) {
        for (Edge<LongWritable, LongWritable> edge : getEdges()) {
            context.write("edge", getId(), edge.getDestVertexId(),
                edge.getValue());
        }
    }
    /**
     * Perform only one iteration.
     **/
    voteToHalt();
}
/**
 * @param args
 * @throws IOException
 */
public static void main(String[] args) throws IOException {
    if (args.length < 4) {
        throw new IOException(
            "Usage: VertexInputFormat <vertex input> <edge input> <vertex output> <edge output>");
    }
    /**
     * Use GraphJob to configure a Graph job.
     */
    GraphJob job = new GraphJob();
    /**
     * 1. Specify input graph data and the table that stores information about edges.
     */
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addInput(TableInfo.builder().tableName(args[1]).build());
    job.set(EDGE_TABLE, args[1]);
    /**
     * 2. Specify the data loading mode and resolve the records to edges. This process is similar to the map stage. The
     generated key is the vertex ID, and the generated value is the edge.
     */
    job.setGraphLoaderClass(VertexInputLoader.class);
    /**
     * 3. Specify the data loading phase to generate the vertex that is used in computing. This process is similar to the
     reduce stage. In the reduce stage, edges that are generated in the map stage are combined into a vertex.
     */
    job.setLoadingVertexResolverClass>LoadingResolver.class);

```

```
job.setVertexResolverClass(LoadVertexResolver.class);  
/**  
 * 4. Specify the actions of the vertex that is used in computing. The vertex.compute() method is used for each iteration.  
 */  
job.setVertexClass(MyVertex.class);  
/**  
 * 5. Specify the result table of the Graph job and write the computing results to the result table.  
 */  
job.addOutput(TableInfo.builder().tableName(args[2]).label("vertex").build());  
job.addOutput(TableInfo.builder().tableName(args[3]).label("edge").build());  
/**  
 * 6. Submit the job for execution.  
 */  
job.run();  
}  
}
```

1.9.6.11. Vertex table import

This topic provides an example of importing a vertex table.

Example:

```
import java.io.IOException;  
import com.aliyun.odps.conf.Configuration;  
import com.aliyun.odps.data.TableInfo;  
import com.aliyun.odps.graph.ComputeContext;  
import com.aliyun.odps.graph.GraphJob;  
import com.aliyun.odps.graph.GraphLoader;  
import com.aliyun.odps.graph.Vertex;  
import com.aliyun.odps.graph.VertexResolver;  
import com.aliyun.odps.graph.MutationContext;  
import com.aliyun.odps.graph.VertexChanges;  
import com.aliyun.odps.graph.Edge;  
import com.aliyun.odps.io.LongWritable;  
import com.aliyun.odps.io.WritableComparable;  
import com.aliyun.odps.io.WritableRecord;  
/**  
 * This example describes how to compile a Graph job program to load data of different data types. It covers how GraphLoader and VertexResolver are used together to build a graph.  
 * A MaxCompute Graph job uses MaxCompute tables as the input. For example, a job uses two tables as the input. One stores information about vertices, and the other stores information about edges.  
 * Format of the table that stores information about vertices:  
 * +-----+  
 * | VertexID | VertexValue |  
 * +-----+  
 * | id0| 9|  
 * +-----+  
 * | id1| 7|  
 * +-----+  
 * | id2| 8|  
 * +-----+  
 *  
 * Format of the table that stores information about edges:  
 * +-----+  
 * | VertexID | DestVertexID| EdgeValue|  
 * +-----+  
 *
```

```

* +-----+
* | id0| id1| 1|
* +-----+
* | id0| id2| 2|
* +-----+
* | id2| id1| 3|
* +-----+
*
* The two tables show that id0 has two outgoing edges that point to id1 and id2. id2 has one outgoing edge that points to id1, and id1 has no outgoing edges.
*
* For data of this type, in GraphLoader::load(LongWritable, Record, MutationContext),
* MutationContext#addVertexRequest(Vertex) can be used to add vertices to the graph.
* link MutationContext#addEdgeRequest(WritableComparable, Edge) can be used to add edges to the graph. In
* link VertexResolver#resolve(WritableComparable, Vertex, VertexChanges, boolean)
* vertices and edges added by using the load() method are combined to a vertex object. This object is used as the return value and added to the graph for computing.
*
**/
public class VertexInputFormat {
    private final static String EDGE_TABLE = "edge.table";
    /**
     * Resolve records to vertices and edges. Each record indicates a vertex or edge based on its source.
     *
     * The following process is similar to the map stage of com.aliyun.odps.mapreduce.Mapper. Enter a record to generate key-value pairs.
     * The keys are vertex IDs, and the values are vertices or edges that are written based on context. These key-value pairs are aggregated by using LoadingVertexResolver based on the vertex IDs.
     *
     * Note: The vertices or edges added here are requests sent based on the record content and are not used in computing.
     * Only the vertices or edges added by using VertexResolver are used for computing.
     */
    public static class VertexInputLoader extends
        GraphLoader<LongWritable, LongWritable, LongWritable, LongWritable> {
        private boolean isEdgeData;
        /**
         * Configure VertexInputLoader.
         *
         * @param conf
         * Indicates the configured parameters of a job. These parameters are configured in the Main function of GraphJob or by running the SET command in the console.
         * @param workerId
         * Indicates the serial number of the running worker. The number starts from 0 and can be used to build a unique vertex ID.
         * @param inputTableInfo
         * Indicates the information about the input table that is loaded to the running worker. The information can be used to determine the data type of the input data (the format of the record).
         */
        @Override
        public void setup(Configuration conf, int workerId, TableInfo inputTableInfo) {
            isEdgeData = conf.get(EDGE_TABLE).equals(inputTableInfo.getTableName());
        }
        /**
         * Resolve the record to edges based on the record content and send a request to add them to the graph.
         *
         * @param recordNum
         * Indicates the serial number of the record. The number starts from 1 and is separately counted in each work

```

```
er.  
 * @param record  
 * Indicates the records in the input table. The table contains three columns, which indicate the source vertex,  
 destination vertex, and edge weight.  
 * @param context  
 * Indicates the context. The context is used when you send a request to add resolved edges to the graph.  
 **/  
@Override  
public void load(  
    LongWritable recordNum,  
    WritableRecord record,  
    MutationContext<LongWritable, LongWritable, LongWritable, LongWritable> context)  
    throws IOException {  
    if (isEdgeData) {  
        /**  
         * Data comes from the table that stores information about edges.  
         *  
         * 1. The first column indicates the IDs of source vertices.  
         */  
        LongWritable sourceVertexID = (LongWritable) record.get(0);  
        /**  
         * 2. The second column indicates the IDs of destination vertices.  
         */  
        LongWritable destinationVertexID = (LongWritable) record.get(1);  
        /**  
         * 3. The third column indicates edge weights.  
         */  
        LongWritable edgeValue = (LongWritable) record.get(2);  
        /**  
         * 4. Create an edge based on a destination vertex ID and an edge weight.  
         */  
        Edge<LongWritable, LongWritable> edge = new Edge<LongWritable, LongWritable>(  
            destinationVertexID, edgeValue);  
        /**  
         * 5. Send a request to add the edge to a source vertex.  
         */  
        context.addEdgeRequest(sourceVertexID, edge);  
        /**  
         * 6. If each record indicates a bidirectional edge, repeat Step 4 and Step 5.  
         * Edge<LongWritable, LongWritable> edge2 = new  
         * Edge<LongWritable, LongWritable>( sourceVertexID, edgeValue);  
         * context.addEdgeRequest(destinationVertexID, edge2);  
         */  
    } else {  
        /**  
         * Data comes from the table that stores information about vertices.  
         *  
         * 1. The first column indicates the IDs of vertices.  
         */  
        LongWritable vertexID = (LongWritable) record.get(0);  
        /**  
         * 2. The second column indicates the values of vertices.  
         */  
        LongWritable vertexValue = (LongWritable) record.get(1);  
        /**  
         * 3. Create a vertex based on an ID and a value.  
         */  
    }  
}
```

```

MyVertex vertex = new MyVertex();
/**
 * 4. Initialize the vertex.
 */
vertex.setId(vertexID);
vertex.setValue(vertexValue);
/**
 * 5. Send a request to add the vertex.
 */
context.addVertexRequest(vertex);
}
}
}
/**
 * Summarize key-value pairs generated by using GraphLoader::load(LongWritable, Record, MutationContext).
 * This process is similar to the reduce stage of com.aliyun.odps.mapreduce.Reducer. For a unique vertex ID, all actions, such as
 * adding or removing vertices or edges for the ID, are stored in VertexChanges.
 *
 * Note: Not only conflicting vertices or edges added by using the load() method are called. A conflict occurs when multiple same vertex objects or duplicate edges are added.
 * All the IDs that are requested to be generated by using the load() method are called.
 */
public static class LoadingResolver extends
    VertexResolver<LongWritable, LongWritable, LongWritable, LongWritable> {
/**
 * Process a request to add or remove vertices or edges for an ID.
 *
 * VertexChanges has four APIs, which correspond to the four APIs of MutationContext:
 * VertexChanges::getAddedVertexList() corresponds to
 * MutationContext::addVertexRequest(Vertex).
 * In the load() method, if a request is sent to add vertex objects with the same ID, the objects are collected to the returned list.
 * VertexChanges::getAddedEdgeList() corresponds to
 * MutationContext::addEdgeRequest(WritableComparable, Edge)
 * If a request is sent to add edge objects with the same source vertex ID, the objects are collected to the returned list.
 * VertexChanges::getRemovedVertexCount() corresponds to
 * MutationContext::removeVertexRequest(WritableComparable)
 * If a request is sent to remove vertices with the same ID, the number of total removal requests is returned.
 * VertexChanges#getRemovedEdgeList() corresponds to
 * MutationContext#removeEdgeRequest(WritableComparable, WritableComparable)
 * If a request is sent to remove edge objects with the same source vertex ID, the objects are collected to the returned list.
 *
 * You can process the changes in the ID and state whether the ID is used in computing in the return value.
 * If the returned vertex is not null, the ID is used in subsequent computing. If the returned vertex is null, the ID is not used in subsequent computing.
 *
 * @param vertexId
 * Indicates the ID of the vertex that is requested to be added or the source vertex ID of the edge that is requested to be added.
 * @param vertex
 * Indicates an existing vertex object. The value of this parameter is always null in the data loading phase.
 * @param vertexChanges
 * Indicates a collection of vertices or edges that are requested to be added or removed for the ID.
 * @param hasMessages
 * Indicates whether messages are sent to the ID. The value of this parameter is always false in the data loading phase.

```

```
g phase.
/**
 * @Override
 * public Vertex<LongWritable, LongWritable, LongWritable, LongWritable> resolve(
 *     LongWritable vertexId,
 *     Vertex<LongWritable, LongWritable, LongWritable, LongWritable> vertex,
 *     VertexChanges<LongWritable, LongWritable, LongWritable, LongWritable> vertexChanges,
 *     boolean hasMessages) throws IOException {
 * /**
 * * 1. Obtain the vertex object for computing.
 * **/
 * MyVertex computeVertex = null;
 * if (vertexChanges.getAddedVertexList() == null
 *     || vertexChanges.getAddedVertexList().isEmpty()) {
 *     computeVertex = new MyVertex();
 *     computeVertex.setId(vertexId);
 * } else {
 * /**
 * * Each record indicates a unique vertex in the table that stores information about vertices.
 * **/
 * computeVertex = (MyVertex) vertexChanges.getAddedVertexList().get(0);
 * }
 * /**
 * * 2. Add the edge, which is requested to be added to the vertex, to the vertex object. If the data is a possible duplicate, deduplicate it based on algorithm requirements.
 * **/
 * if (vertexChanges.getAddedEdgeList() != null) {
 *     for (Edge<LongWritable, LongWritable> edge : vertexChanges
 *         .getAddedEdgeList()) {
 *         computeVertex.addEdge(edge.getDestVertexId(), edge.getValue());
 *     }
 * }
 * /**
 * * 3. Return the vertex object and add it to the final graph for computing.
 * **/
 * return computeVertex;
 * }
 * }
 * /**
 * * Determine the actions of the vertex that is used in computing.
 * *
 * **/
 * public static class MyVertex extends
 *     Vertex<LongWritable, LongWritable, LongWritable, LongWritable> {
 * /**
 * * Write the edge of the vertex to the result table based on the format of the input table. Make sure that the formats and data of the input and output tables are the same.
 * *
 * * @param context
 * *     Indicates the runtime context.
 * * @param messages
 * *     Indicates the input messages.
 * **/
 * @Override
 * public void compute(
 *     ComputeContext<LongWritable, LongWritable, LongWritable, LongWritable> context,
 *     Iterable<LongWritable> messages) throws IOException {
 * /**
```

```

/**
 * Write the ID and value of the vertex to the result table that stores information about vertices.
 **/
context.write("vertex", getId(), getValue());
/**
 * Write the edge of the vertex to the result table that stores information about edges.
 **/
if (hasEdges()) {
    for (Edge<LongWritable, LongWritable> edge : getEdges()) {
        context.write("edge", getId(), edge.getDestVertexId(),
            edge.getValue());
    }
}
/**
 * Perform only one iteration.
 **/
voteToHalt();
}
}
/**
 * @param args
 * @throws IOException
 */
public static void main(String[] args) throws IOException {
    if (args.length < 4) {
        throw new IOException(
            "Usage: VertexInputFormat <vertex input> <edge input> <vertex output> <edge output>");
    }
    /**
     * Use GraphJob to configure a Graph job.
     */
    GraphJob job = new GraphJob();
    /**
     * 1. Specify input graph data and the table that stores information about edges.
     */
    job.addInput(TableInfo.builder().tableName(args[0]).build());
    job.addInput(TableInfo.builder().tableName(args[1]).build());
    job.set(EDGE_TABLE, args[1]);
    /**
     * 2. Specify the data loading mode and resolve the records to edges. This process is similar to the map stage. The
     generated key is the vertex ID, and the generated value is the edge.
     */
    job.setGraphLoaderClass(VertexInputLoader.class);
    /**
     * 3. Specify the data loading phase to generate the vertex that is used in computing. This process is similar to the
     reduce stage. In the reduce stage, edges that are generated in the map stage are combined into a vertex.
     */
    job.setLoadingVertexResolverClass>LoadingResolver.class);
    /**
     * 4. Specify the actions of the vertex that is used in computing. The vertex.compute() method is used for each iter
     ation.
     */
    job.setVertexClass(MyVertex.class);
    /**
     * 5. Specify the result table of the Graph job and write the computing results to the result table.
     */
    job.addOutput(TableInfo.builder().tableName(args[2]).label("vertex").build());
    job.addOutput(TableInfo.builder().tableName(args[3]).label("edge").build());
}

```

```
/**
 * 6. Submit the job for execution.
 */
job.run();
}
```

1.10. Java SDK

This topic describes MaxCompute SDK for Java.

MaxCompute SDK for Java provides a variety of APIs to support development on MaxCompute.

For more information about the APIs, see *MaxCompute Developer Guide*.

1.11. PyODPS

1.11.1. Quick start

PyODPS is MaxCompute SDK for Python. It provides easy-to-use Python programming interfaces. Similar to pandas, PyODPS provides fast, flexible, and expressive data structures. You can use the data processing feature of PyODPS, which is similar to that of pandas, by calling the DataFrame API provided by PyODPS. This topic describes how to use PyODPS in your projects.

Background information

You can develop most programs in MaxCompute by using SQL statements. However, you must use Python to develop complex business logic and user defined functions (UDFs). For example, you must use Python in the following scenarios:

- **Interface connection:** An external service must provide required information to complete authentication before the service can access any data record in MaxCompute tables by using HTTP interfaces.
- **Asynchronous invocation:** In most cases, the system creates a node for each of thousands of tasks with similar data processing logic. It is difficult to manage these nodes, and excessive resources are occupied at the same time. PyODPS allows you to use queues to asynchronously run SQL tasks at a high concurrency. Queues also help you manage all nodes in a centralized manner.
- **UDF development:** If the built-in functions of MaxCompute cannot meet your requirements, you can develop UDFs. For example, you can develop a UDF to format data of the DECIMAL type with thousands separators.
- **SQL performance improvement:** To check whether database transactions follow the first in, first out (FIFO) rule, you must compare each new record with historical records in the only transaction table. In most cases, this transaction table contains a large number of records. If you use SQL statements to complete the comparison, the time complexity is $O(N^2)$. In addition, the SQL statements may fail to return the expected result. To resolve this issue, you can call the cache method in Python code to traverse records in the transaction table only once. In this case, the time complexity is $O(N)$ and the efficiency is significantly improved.
- **Other scenarios:** You can use Python to develop an amount allocation model. For example, you can develop a model to allocate USD 10 to three persons. In this model, you must define the logic for handling the cash over and short.

Procedure

This topic describes how to use a PyODPS node in DataWorks for development.

1. Log on to the DataWorks console.

2. Create a PyODPS node.
3. Edit the PyODPS node.
 - i. Write the code of the PyODPS node.

Write the test code in the code editor of the PyODPS node. In this example, write the following code in the code editor. It covers a full range of table operations.

```
from odps import ODPS
import sys
reload(sys)
# Set UTF-8 as the default encoding format. You must execute this statement if the data contains Chinese characters.
sys.setdefaultencoding('utf8')
# Create a non-partitioned table named my_new_table, which contains the fields with the specified names and of the specified data types.
table = o.create_table('my_new_table', 'num bigint, id string', if_not_exists=True)
# Write data to the my_new_table table.
records = [[111, 'aaa'],
           [222, 'bbb'],
           [333, 'ccc'],
           [444, 'Chinese']]
o.write_table(table, records)
# Read data from the my_new_table table.
for record in o.read_table(table):
    print record[0],record[1]
# Read data from the my_new_table table by executing an SQL statement.
result = o.execute_sql('select * from my_new_table;',hints={'odps.sql.allow.fullscan': 'true'})
# Obtain the execution result of the SQL statement.
with result.open_reader() as reader:
    for record in reader:
        print record[0],record[1]
# Delete the table.
table.drop()
```

- ii. Run the code.

After you write the code, click the  icon. After the code is run, you can view the running result of the PyODPS node on the **Run Log** tab. The result in the following figure indicates that the running is successful.

1.11.2. Installation guide

This topic describes how to install PyODPS.

If you can access the Internet, we recommend that you use the Python package installer pip to install PyODPS. For more information, see [pip installation](#). If you want to speed up the download, we recommend that you use [Alibaba Cloud images](#).

Prerequisites

The following requirements are met before you install PyODPS:

- The setuptools version is 3.0 or later.
- The requests version is 2.4.0 or later.

Installation commands for reference:

```
pip install setuptools>=3.0
pip install requests>=2.4.0
```

Installation suggestions

We recommend that you install the following tools to accelerate Tunnel-based data upload:

- Greenlet. Recommended version: 0.4.10 or later.
- Cython. Recommended version: 0.19.0 or later.

The following installation commands are for your reference:

```
pip install greenlet>=0.4.10 # Optional. It accelerates Tunnel-based data upload.
pip install cython>=0.19.0 # Optional. We recommend that you do not install Cython if you use a Windows operating system.
```

 **Note** If you use a Windows operating system, make sure that you have installed Visual C++ and Cython of correct versions. Otherwise, you cannot accelerate Tunnel-based data upload. For more information about the versions of Visual C++ and Cython, see [WindowsCompilers](#).

Installation procedure

1. Run the following command to install PyODPS:

```
pip install pyodps
```

2. Run the following command to check whether the installation is successful:

```
python -c "from odps import ODPS"
```

3. If the Python version is not the default version, run the following command to switch to the default version after you have installed pip:

```
/home/tops/bin/python2.7 -m pip install setuptools>=3.0
```

1.11.3. Platform instructions

1.11.3.1. Overview

PyODPS can be called as a data development node on a data development platform such as DataWorks. The platform provides a PyODPS running environment and can schedule and run nodes. You do not need to manually create a MaxCompute entry object.

To migrate PyODPS nodes from a data development platform to a manually deployed PyODPS environment, read the instructions in the following topics:

1.11.3.2. Use local PyODPS

If you want to debug PyODPS locally or the resources on the platform where PyODPS is deployed cannot meet your requirements, you can deploy a local PyODPS environment. This topic describes how to deploy a local PyODPS environment.

1. Install PyODPS in a local environment. For more information, see [Installation instructions](#).
2. Create a MaxCompute entry object in the local environment.

You can execute the following statement on the data development platform to generate a template of the statement for creating a MaxCompute entry object. Then, you can modify the template to obtain the required statement.

```
print("\nfrom odps import ODPS\no = ODPS(%r, '<access-key>', %r, '<endpoint>')\n" % (o.account.access_id, o.project))
```

- Place the obtained statement at the beginning of all code.

1.11.3.3. Use PyODPS in DataWorks

This topic describes how to use PyODPS in DataWorks and the limits.

Create a PyODPS node

You can create a PyODPS node for a workflow.

 **Note** For more information, see [Create a PyODPS node](#).

MaxCompute entry

The PyODPS node in DataWorks contains a global variable `odps` or `o`, which is the MaxCompute entry. You do not need to manually define the MaxCompute entry.

```
print(o.exist_table('pyodps_iris'))
```

Execute SQL statements

You can execute SQL statements in the PyODPS node. For more information, see [SQL](#).

By default, InstanceTunnel is disabled in DataWorks, and `instance.open_reader` is executed by using the Result interface. In this case, a maximum of 10,000 data records can be read. You can use `reader.count` to obtain the number of data records. If you need to obtain all data iteratively, you must disable the limit on the data volume. You can execute the following statements to enable InstanceTunnel and disable the limit.

```
options.tunnel.use_instance_tunnel = True
options.tunnel.limit_instance_tunnel = False # Disable the limit on the data volume.
with instance.open_reader() as reader:
    # Use InstanceTunnel to read all data.
```

You can add `tunnel=True` to `open_reader` to enable InstanceTunnel for the current `open_reader` operation. You can add `limit=False` to `open_reader` to disable the limit on the data volume for the current `open_reader` operation.

```
with instance.open_reader(tunnel=True, limit=False) as reader:
    # The current open_reader operation is executed by using InstanceTunnel, and all data can be read.
```

 **Note** If you do not enable InstanceTunnel, the format of the obtained data may be invalid.

DataFrame

- Perform operations on DataFrames.

To perform operations on DataFrames in DataWorks, you must explicitly call automatically executed methods, such as `execute` and `head`.

```
from odps.df import DataFrame
iris = DataFrame(o.get_table('pyodps_iris'))
for record in iris[iris.sepal_width < 3].execute(): # Call an automatically executed method to process each data record.
```

To call an automatically executed method for data display, set `options.interactive` to True.

```
from odps import options
from odps.df import DataFrame
options.interactive = True # Set options.interactive to True at the beginning of the code.
iris = DataFrame(o.get_table('pyodps_iris'))
print(iris.sepal_width.sum()) # The method is executed immediately when the system displays information.
```

- Display details.

To display details, you must set `options.verbose` to True. By default, this parameter is set to True in DataWorks. The system displays details such as the Logview URL during the running process.

Obtain scheduling parameters

Different from SQL nodes in DataWorks, a PyODPS node does not replace strings such as `${param_name}` in the code. Instead, it adds a dictionary named `args` as a global variable before it runs the code. You can obtain the scheduling parameters from the dictionary. This way, the Python code is not affected. For example, on the **Scheduling configuration** tab of a PyODPS node in DataWorks, you can specify `ds=${yyyymmdd}` in the **Parameters** field in the Basic properties section. Then, you can specify the following commands in the code of the node to obtain the parameter value:

```
print('ds=' + args['ds'])
ds=yyyymmdd
```

 **Note** You can run the following command to obtain the partition named `ds=${yyyymmdd}`:

```
o.get_table('table_name').get_partition('ds=' + args['ds'])
```

Limits on functions

- The Python version of a PyODPS node is 2.7.
- Each PyODPS node can process a maximum of 50 MB data and can occupy a maximum of 1 GB memory. Otherwise, DataWorks terminates the PyODPS node. Do not write unnecessary Python data processing code in PyODPS nodes.
- Writing and debugging code in DataWorks is inefficient. We recommend that you install an IDE locally to write code.
- To avoid excess pressure on the gateway of DataWorks, DataWorks limits the CPU utilization and memory usage. If the system displays Got killed, the memory usage exceeds the limit, and the system terminates the related processes. Therefore, we recommend that you do not perform local data operations. However, the limits on the memory usage and CPU utilization do not apply to SQL or DataFrame nodes, except to `_pandas`, that are initiated by PyODPS.
- Functions may be limited in the following aspects due to the lack of packages such as `matplotlib`:
 - The use of the `plot` function of DataFrame is affected.
 - DataFrame user defined functions (UDFs) can be executed only after they are submitted to MaxCompute. As required by the Python sandbox, you can only use pure Python libraries and the NumPy library to execute UDFs. Other third-party libraries such as `pandas` cannot be used.

- However, you can use the NumPy and pandas libraries that are pre-installed in DataWorks to execute non-UDFs. You are not allowed to use other third-party libraries that contain binary code.
- For compatibility reasons, `options.tunnel.use_instance_tunnel` is set to False in DataWorks by default. If you want to enable `InstanceTunnel` globally, you must set this parameter to True.
- For implementation reasons, the Python atexit package is not supported. You must use the try-finally structure to implement related features.

1.11.4. Basic operations

1.11.4.1. Overview

PyODPS supports basic operations on MaxCompute objects. You can use Python-compliant programming methods to perform operations on MaxCompute.

PyODPS allows you to perform basic operations on the following MaxCompute objects: projects, tables, SQL, task instances, resources, and functions.

1.11.4.2. Projects

This topic describes how to perform basic operations on projects by using PyODPS.

You can perform the following basic operations for a project:

- Obtain a project. You can call the `get_project()` method of a MaxCompute entry object to obtain a specific project.

```
project = o.get_project('my_project') # Obtain the specified project.
project = o.get_project()             # Obtain the current project.
```

The method requires a project name. If you do not specify a project name for the method, the method returns the current project.

- Check whether a project exists. You can call the `exist_project()` method to check whether a specific project exists.

1.11.4.3. Tables

This topic describes how to perform basic operations on tables by using PyODPS.

Basic operations

You can perform the following basic operations on tables:

- Obtain all tables in the current project by calling the `list_tables()` method of a MaxCompute entry object.

```
# Obtain all tables in the current project.
for table in o.list_tables():
```

- Check whether a specific table exists by calling the `exist_table()` method of a MaxCompute entry object.
- Obtain a specific table by calling the `get_table()` method of a MaxCompute entry object.

```
t = o.get_table('table_name')
t.schema
odps.Schema {
  c_int_a      bigint
  c_int_b      bigint
  c_double_a   double
  c_double_b   double
  c_string_a   string
  c_string_b   string
  c_bool_a     boolean
  c_bool_b     boolean
  c_datetime_a datetime
  c_datetime_b datetime
}
t.lifecycle
-1
print(t.creation_time)
2014-05-15 14:58:43
t.is_virtual_view
False
t.size
1408
t.comment
'table_name Table Comment'
t.schema.columns
[<column c_int_a, type bigint>,
 <column c_int_b, type bigint>,
 <column c_double_a, type double>,
 <column c_double_b, type double>,
 <column c_string_a, type string>,
 <column c_string_b, type string>,
 <column c_bool_a, type boolean>,
 <column c_bool_b, type boolean>,
 <column c_datetime_a, type datetime>,
 <column c_datetime_b, type datetime>]
t.schema['c_int_a']
<column c_int_a, type bigint>
t.schema['c_int_a'].comment
'Comment of column c_int_a'
```

You can obtain a table from another project by specifying the project parameter.

```
t = o.get_table('table_name', project='other_project')
```

Create a table schema

You can use one of the following methods to create a table schema:

- Create a schema based on table columns and optional partitions.

```

from odps.models import Schema, Column, Partition
columns = [Column(name='num', type='bigint', comment='the column'),
           Column(name='num2', type='double', comment='the column2')]
partitions = [Partition(name='pt', type='string', comment='the partition')]
schema = Schema(columns=columns, partitions=partitions)
schema.columns
[<column num, type bigint>,
 <column num2, type double>,
 <partition pt, type string>]
schema.partitions
[<partition pt, type string>]
schema.names # Obtain the names of non-partition fields.
['num', 'num2']
schema.types # Obtain the data types of non-partition fields.
[bigint, double]

```

- Create a schema by calling the `Schema.from_lists()` method. This method is more convenient, but you cannot directly set comments for columns and partitions.

```

schema = Schema.from_lists(['num', 'num2'], ['bigint', 'double'], ['pt'], ['string'])
schema.columns
[<column num, type bigint>,
 <column num2, type double>,
 <partition pt, type string>]

```

Create a table

You can call the `create_table()` method to create a table in two ways:

- Use a table schema to create a table.

```

table = o.create_table('my_new_table', schema)
table = o.create_table('my_new_table', schema, if_not_exists=True) # Create the table only if no table with the same name exists.
table = o.create_table('my_new_table', schema, lifecycle=7) # Set the lifecycle of the table.

```

- Create a table by specifying the names and data types of the fields to be contained in the table.

```

# Create a non-partitioned table.
table = o.create_table('my_new_table', 'num bigint, num2 double', if_not_exists=True)
# Create a partitioned table with the specified table columns and partition fields.
table = o.create_table('my_new_table', ('num bigint, num2 double', 'pt string'), if_not_exists=True)

```

By default, when you create a table, you can use only the `BIGINT`, `DOUBLE`, `DECIMAL`, `STRING`, `DATETIME`, `BOOLEAN`, `MAP`, and `ARRAY` data types. If you need to use other data types such as `TINYINT` and `STRUCT`, you must set `options.sql.use_odps2_extension` to `True`. Example:

```

from odps import options
options.sql.use_odps2_extension = True
table = o.create_table('my_new_table', 'cat smallint, content struct<title:varchar(100), body:string>')

```

Synchronize table updates

After another program updates a table, for example, the table schema, you can call the `reload()` method to synchronize the update.

```
table.reload()
```

Insert a record to a table

A record refers to a single row in a table. You can call the `new_record()` method to insert a record to a specific table.

```
t = o.get_table('mytable')
r = t.new_record(['val0', 'val1']) # The number of values must be equal to the number of fields in the table schema.
r2 = t.new_record() # You can leave the value empty.
r2[0] = 'val0' # Set a value based on an offset.
r2['field1'] = 'val1' # Set a value based on the field name.
r2.field1 = 'val1' # Set a value based on an attribute.
print(record[0]) # Obtain the value at position 0.
print(record['c_double_a']) # Obtain a value based on a field.
print(record.c_double_a) # Obtain a value based on an attribute.
print(record[0:3]) # Perform slicing operations.
print(record[0, 2, 3]) # Obtain values at multiple positions.
print(record['c_int_a', 'c_double_a']) # Obtain values based on multiple fields.
```

Obtain table data

You can use one of the following methods to obtain data from a table:

- Call the `read_table()` method of a MaxCompute entry object to read data from a table.

```
for record in o.read_table('test_table', partition='pt=test'):
    # Process a record.
```

- Call the `head()` method to obtain less than 10,000 data records from the beginning of a table.

```
t = o.get_table('table_name')
# Process each record.
for record in t.head(3):
```

- Call the `open_reader()` method.
 - Open the reader with a WITH clause.

```
with t.open_reader(partition='pt=test') as reader:
    count = reader.count
    for record in reader[5:10] # You can execute the statement multiple times until all records are read. The number of records is specified by count. You can change the code to parallel-operation code.
        # Process a record.
```

- Open the reader without a WITH clause.

```
reader = t.open_reader(partition='pt=test')
count = reader.count
for record in reader[5:10] # You can execute the statement multiple times until all records are read. The number of records is specified by count. You can change the code to parallel-operation code.
    # Process a record.
```

Write data to a table

- Call the `write_table()` method of a MaxCompute entry object to write data to a table.

```
records = [[111, 'aaa', True],      # A list can be specified.
           [222, 'bbb', False],
           [333, 'ccc', True],
           [444, 'Chinese', False]]
o.write_table('test_table', records, partition='pt=test', create_partition=True)
```

Note

- Each time you call the `write_table()` method, MaxCompute generates a file on the server. This operation is time-consuming. In addition, if an excessive number of files are generated, the efficiency of subsequent queries is affected. We recommend that you write multiple records at a time or provide a generator object if you use the `write_table()` method.
- If you call the `write_table()` method to write data to a table, new data will be appended to existing data. PyODPS does not provide options to overwrite existing data. You must manually delete the data that you want to overwrite. For a non-partitioned table, you must call the `table.truncate()` method to delete data. For a partitioned table, you must delete partitions first and then create partitions again.

- Call the `open_writer()` method to write data to a table.

```
with t.open_writer(partition='pt=test') as writer:
records = [[111, 'aaa', True],      # A list can be specified.
           [222, 'bbb', False],
           [333, 'ccc', True],
           [444, 'Chinese', False]]
writer.write(records) # records can be iterable objects.
with t.open_writer(partition='pt1=test1,pt2=test2') as writer: # Write data in multi-level partitioning mode.
records = [t.new_record([111, 'aaa', True]), # Record objects can be used.
           t.new_record([222, 'bbb', False]),
           t.new_record([333, 'ccc', True]),
           t.new_record([444, 'Chinese', False])]
writer.write(records)
```

If the specified partition does not exist, set the `create_partition` parameter to `True` to create a partition.
Example:

```
with t.open_writer(partition='pt=test', create_partition=True) as writer:
records = [[111, 'aaa', True],      # A list can be specified.
           [222, 'bbb', False],
           [333, 'ccc', True],
           [444, 'Chinese', False]]
writer.write(records) # records can be iterable objects.
```

- Use multiple processes to concurrently write data to a table.

If multiple processes concurrently write data to a table, all processes use the same session ID but write data to different blocks. Each block corresponds to a file on the server. After all the processes finish writing data, the main process submits the data.

```
import random
from multiprocessing import Pool
from odps.tunnel import TableTunnel
def write_records(session_id, block_id):
    # Create a session with the specified session ID.
    local_session = tunnel.create_upload_session(table.name, upload_id=session_id)
    # Create a writer with the specified block ID.
    with local_session.open_record_writer(block_id) as writer:
        for i in range(5):
            # Generate data and write the data to the correct block.
            record = table.new_record([random.randint(1, 100), random.random()])
            writer.write(record)
if __name__ == '__main__':
    N_WORKERS = 3
    table = o.create_table('my_new_table', 'num bigint, num2 double', if_not_exists=True)
    tunnel = TableTunnel(o)
    upload_session = tunnel.create_upload_session(table.name)
    # All processes use the same session ID.
    session_id = upload_session.id
    pool = Pool(processes=N_WORKERS)
    futures = []
    block_ids = []
    for i in range(N_WORKERS):
        futures.append(pool.apply_async(write_records, (session_id, i)))
        block_ids.append(i)
    [f.get() for f in futures]
    # Submit the data in all the blocks.
    upload_session.commit(block_ids)
```

Delete a table

You can call the `delete_table()` method to delete an existing table.

```
o.delete_table('my_table_name', if_exists=True) # Delete a table only if the table exists.
t.drop() # Call the drop() method to delete a table if the table exists.
```

Create a DataFrame

PyODPS provides a DataFrame framework, which allows you to conveniently query and manage MaxCompute data. You can call the `to_df()` method to convert a table to a DataFrame.

```
table = o.get_table('my_table_name')
df = table.to_df()
```

Manage partitions

- Check whether a table is partitioned.

```
if table.schema.partitions:
    print('Table %s is partitioned.' % table.name)
```

- Iterate over all the partitions in a table.

```
for partition in table.partitions:
    print(partition.name)
for partition in table.iterate_partitions(spec='pt=test'):
    # Iterate over level-2 partitions.
```

- Check whether a partition exists.

```
table.exist_partition('pt=test,sub=2015')
```

- Obtain a partition.

```
partition = table.get_partition('pt=test')
print(partition.creation_time)
2015-11-18 22:22:27
partition.size
0
```

- Create a partition.

```
t.create_partition('pt=test', if_not_exists=True) # Create a partition only if no partition with the same name exists.
```

- Delete a partition.

```
t.delete_partition('pt=test', if_exists=True) # Delete a partition only if the partition exists.
partition.drop() # Call the drop() method to delete a partition if the partition exists.
```

Upload and download data by using MaxCompute Tunnel

MaxCompute Tunnel is the data tunnel of MaxCompute. You can use Tunnel to upload data to or download data from MaxCompute.

 **Note** We recommend that you use the write and read interfaces of tables instead of Tunnel. In a Cython environment, PyODPS compiles C programs during installation to accelerate the Tunnel upload and download.

- Example of data upload

```
from odps.tunnel import TableTunnel
table = o.get_table('my_table')
tunnel = TableTunnel(odps)
upload_session = tunnel.create_upload_session(table.name, partition_spec='pt=test')
with upload_session.open_record_writer(0) as writer:
    record = table.new_record()
    record[0] = 'test1'
    record[1] = 'id1'
    writer.write(record)
    record = table.new_record(['test2', 'id2'])
    writer.write(record)
upload_session.commit([0])
```

- Example of data download

```
from odps.tunnel import TableTunnel
tunnel = TableTunnel(odps)
download_session = tunnel.create_download_session('my_table', partition_spec='pt=test')
with download_session.open_record_reader(0, download_session.count) as reader:
    for record in reader:
        # Process each record.
```

 **Note** PyODPS does not support the upload of external tables.

1.11.4.4. SQL

This topic describes how to perform basic operations related to SQL statements by using PyODPS.

PyODPS supports MaxCompute SQL queries and provides methods to obtain execution results.

Execute SQL statements

You can call the `execute_sql()` and `run_sql()` methods of a MaxCompute entry object to execute SQL statements for creating task instances.

Parameter description:

- `statement`: the SQL statement to execute.
- `hints`: the runtime parameters. The `hints` parameter is of the DICT type.

Examples:

- Execute SQL statements.

```
o.execute_sql('select * from dual') # Execute the statement in synchronous mode. Other instances are blocked until the SQL statement is executed.
instance = o.run_sql('select * from dual') # Execute the statement in asynchronous mode.
print(instance.get_logview_address()) # Obtain the Logview URL of an instance.
instance.wait_for_success() # Other instances are blocked until the SQL statement is executed.
```

- Set runtime parameters for SQL statements.

```
o.execute_sql('select * from pyodps_iris', hints={'odps.sql.mapper.split.size': 16})
```

If you set the `sql.settings` parameter globally, you need to configure runtime parameters each time you execute the statement.

```
from odps import options
options.sql.settings = {'odps.sql.mapper.split.size': 16}
o.execute_sql('select * from pyodps_iris') # The hints parameter is automatically set based on global settings.
```

Obtain the execution results of SQL statements

You can call the `open_reader` method to obtain execution results of SQL statements. When the query results are being read, the following situations may occur:

- The SQL statements return structured data.

```
with o.execute_sql('select * from dual').open_reader() as reader:
    for record in reader:
        # Process each record.
```

- If the `DESC` command is executed, you can use `reader.raw` to obtain the raw SQL query results.

```
with o.execute_sql('desc dual').open_reader() as reader:
    print(reader.raw)
```

If you specify `options.tunnel.use_instance_tunnel=True` when you call the `open_reader` method, PyODPS calls `InstanceTunnel` by default. However, if you are using an earlier version of MaxCompute or an error occurs when PyODPS calls `InstanceTunnel`, PyODPS generates an alert and automatically downgrades the call object to the old `Result` interface. You can identify the cause of the downgrade based on the alert information. If the result of `InstanceTunnel` does not meet your expectation, set this option to `False`. When you call the `open_reader` method, you can specify whether to call `InstanceTunnel` or the `Result` interface by setting the `tunnel` parameter.

```
# Call Instance Tunnel.
with o.execute_sql('select * from dual').open_reader(tunnel=True) as reader:
    for record in reader:
        # Process each record.
# Call the Result interface.
with o.execute_sql('select * from dual').open_reader(tunnel=False) as reader:
    for record in reader:
        # Process each record.
```

By default, PyODPS does not limit the amount of data that can be read from an instance. For protected projects, the amount of data that can be downloaded by using `Tunnel` is limited. If `options.tunnel.limit_instance_tunnel` is not set, the limit is automatically enabled. The number of data entries that can be downloaded is limited based on project configurations. A maximum of 10,000 data entries can be downloaded. You can add a `limit` option to the `open_reader` method or set `options.tunnel.limit_instance_tunnel` to `True` to manually limit the amount.

If your MaxCompute version only supports the old `Result` interface and you need to read all data, you can export the execution result of the SQL statement to another table and then use the table read interface to read data. This may be limited by project security settings.

In PyODPS V0.7.7.1 and later, you can use the `open_reader` method to call `InstanceTunnel` to obtain all data.

```
instance = o.execute_sql('select * from movielens_ratings limit 20000')
with instance.open_reader() as reader:
    print(reader.count)
    # for record in reader: Traverse the 20,000 data records. In this example, only 10 data records are obtained based
    on data slicing.
    for record in reader[:10]:
        print(record)
```

Set an alias

If the resource referenced by a UDF changes dynamically, you can set an alias for the old resource and use the alias as the name of the new resource. This way, you do not need to delete the UDF or create another UDF.

```
from odps.models import Schema
myfunc = ""\
from odps.udf import annotate
from odps.distcache import get_cache_file
@annotate('bigint->bigint')
class Example(object):
    def __init__(self):
        self.n = int(get_cache_file('test_alias_res1').read())
    def evaluate(self, arg):
        return arg + self.n
""
res1 = o.create_resource('test_alias_res1', 'file', file_obj='1')
o.create_resource('test_alias.py', 'py', file_obj=myfunc)
o.create_function('test_alias_func',
                  class_type='test_alias.Example',
                  resources=['test_alias.py', 'test_alias_res1'])
table = o.create_table(
    'test_table',
    schema=Schema.from_lists(['size'], ['bigint']),
    if_not_exists=True
)
data = [[1, ], ]
# Write a row of data that contains only one value: 1.
o.write_table(table, 0, [table.new_record(it) for it in data])
with o.execute_sql(
    'select test_alias_func(size) from test_table').open_reader() as reader:
    print(reader[0][0])
res2 = o.create_resource('test_alias_res2', 'file', file_obj='2')
# Set the alias of resource res1 as the name of resource res2 without the need to modify the UDF or resource.
with o.execute_sql(
    'select test_alias_func(size) from test_table',
    aliases={'test_alias_res1': 'test_alias_res2'}).open_reader() as reader:
    print(reader[0][0])
```

Execute SQL statements in an interactive environment

In IPython and Jupyter, SQL plug-ins can be used to execute SQL statements and parameterized queries are supported.

Set biz_id

Occasionally, you must specify biz_id for the SQL statement to execute. Otherwise, an error occurs. In this case, you can set biz_id in the global options to troubleshoot the error.

```
from odps import options
options.biz_id = 'my_biz_id'
o.execute_sql('select * from pyodps_iris')
```

1.11.4.5. Task instances

This topic describes how to perform basic operations on task instances by using PyODPS.

Basic operations

Tasks are implemented as MaxCompute instances. You can call `list_instances` to retrieve all the instances in the project. You can use `exist_instance` to determine if an instance exists, and use `get_instance` to retrieve instances.

```
for instance in o.list_instances():
    print(instance.id)
    o.exist_instance('my_instance_id')
```

You can call the `stop` method for an instance or call `stop_instance` at the MaxCompute entry to stop an instance.

Obtain the Logview URL of an instance

For an SQL task instance, you can call the `get_logview_address` method to obtain its Logview URL.

```
# Obtain the Logview URL based on an existing instance object.
instance = o.run_sql('desc pyodps_iris')
print(instance.get_logview_address())
# Obtain the Logview URL based on an instance ID.
instance = o.get_instance('2016042605520945g9k5pvvi2')
print(instance.get_logview_address())
```

For an XFlow task instance, you must enumerate its sub-instances and obtain the Logview URL of each sub-instance.

```
instance = o.run_xflow('AppendID', 'algo_public', {'inputTableName': 'input_table', 'outputTableName': 'output_table'})
for sub_inst_name, sub_inst in o.get_xflow_sub_instances(instance).items():
    print('%s: %s' % (sub_inst_name, sub_inst.get_logview_address()))
```

Obtain the status of an instance

An instance can be in the Running, Suspended, or Terminated state. You can obtain the status of an instance from the `status` property. You can call the `is_terminated` method to check whether the execution of the current instance is complete and call the `is_successful` method to check whether the execution is successful. If the task is running or fails, `False` is returned for both methods.

```
instance = o.get_instance('2016042605520945g9k5pvvi2')
instance.status
<Status.TERMINATED: 'Terminated'>
from odps.models import Instance
instance.status == Instance.Status.TERMINATED
True
instance.status.value
'Terminated'
```

You can call the `wait_for_completion` method to ask the system to return the status until the execution of the instance is completed. The `wait_for_success` method functions similarly. The difference is that if the execution fails, an exception is reported.

Perform operations on sub-instances

A running instance may contain one or more sub-instances.

You can call the `get_task_names` method to obtain the names of all sub-instances.

```
instance.get_task_names()
['SQLDropTableTask']
```

After you obtain a sub-instance name, you can call `get_task_result` to obtain the execution result of the sub-instance. This method returns the execution result of each sub-instance in the form of a dictionary.

```
instance = o.execute_sql('select * from pyodps_iris limit 1')
instance.get_task_names()
['AnonymousSQLTask']
instance.get_task_result('AnonymousSQLTask')
''sepallength","sepalwidth","petallength","petalwidth","name"\n5.1,3.5,1.4,0.2,"Iris-setosa"\n'
instance.get_task_results()
OrderedDict([('AnonymousSQLTask', 'sepallength', 'sepalwidth', 'petallength', 'petalwidth', 'name'\n5.1,3.5,1.4,
0.2, 'Iris-setosa'\n'))
```

You can call `get_task_progress` to obtain the current running progress of a specific sub-instance when the instance is running.

```
while not instance.is_terminated():
    for task_name in instance.get_task_names():
        print(instance.id, instance.get_task_progress(task_name).get_stage_progress_formatted_string())
        time.sleep(10)
20190519101349613gzbfzuck2 2019-05-19 18:14:03 M1_Stg1_job0:0/1/1[100%]
```

1.11.4.6. Resources

This topic describes how to perform basic operations on resources by using PyODPS.

PyODPS supports file and table resources.

Resources commonly apply to UDFs and MapReduce on MaxCompute.

You can call the `list_resources` method to query all resources, the `exist_resource` method to check whether a resource exists, and the `delete_resource` method to delete a resource. You can also call the `drop` method to delete a resource.

Manage file resources

PyODPS supports basic file types, and the .py, .jar, and .archive types.

- Create a file resource

You can specify a resource name, a file type, a file-like object or a string object to create a file resource.

```
resource = o.create_resource('test_file_resource', 'file', file_obj=open('/to/path/file')) # Use a file-like object to create a file resource.
resource = o.create_resource('test_py_resource', 'py', file_obj='import this') # Use a string to create a file resource.
```

- Read and modify a file resource

You can call the `open` method for a file resource or call the `open_resource` method at the MaxCompute entry to open a file resource. The opened object is a file-like object. Similar to the `open` method built in Python, file resources also support various opening modes.

```

with resource.open('r') as fp: # Open the specified file in read mode.
    content = fp.read() # Read all content.
    fp.seek(0) # Return to the beginning of the file.
    lines = fp.readlines() # Read multiple lines.
    fp.write('Hello World') # Error. Data cannot be written to a file in read mode.
with o.open_resource('test_file_resource', mode='r+') as fp: # Open the file in read/write mode.
    fp.read()
    fp.tell() # Locate the current position.
    fp.seek(10)
    fp.truncate() # Truncate the file to the specified length.
    fp.writelines(['Hello\n', 'World\n']) # Write multiple lines to the file.
    fp.write('Hello World')
    fp.flush() # Manually call the method to submit the update to MaxCompute.

```

PyODPS supports the following opening modes:

- **r**: read mode. The file can be opened, but data cannot be written to it.
- **w**: write mode. Data can be written to the file, but data in the file cannot be read. If a file is opened in write mode, the file content is cleared first.
- **a**: append mode. Data can be added to the end of the file.
- **r+**: read/write mode. You can read data from and write data to the file.
- **w+**: This mode is similar to the **r+** mode. The only difference is that the file content is cleared first.
- **a+**: This mode is similar to the **r+** mode. The only difference is that data can be written only to the end of the file.

In PyODPS, file resources can be opened in binary mode. For example, some compressed files must be opened in binary mode. **rb** indicates that a file is opened in binary read mode, and **r+b** indicates that a file is opened in binary read/write mode.

Manage table resources

- Create a table resource

```
o.create_resource('test_table_resource', 'table', table_name='my_table', partition='pt=test')
```

- Update a table resource

```
table_resource = o.get_resource('test_table_resource')
table_resource.update(partition='pt=test2', project_name='my_project2')
```

- Obtain a table and a partition

```
table_resource = o.get_resource('test_table_resource')
table = table_resource.table
print(table.name)
partition = table_resource.partition
print(partition.spec)
```

- Read data from and write data to a table

```
table_resource = o.get_resource('test_table_resource')
with table_resource.open_writer() as writer:
    writer.write([0, 'aaaa'])
    writer.write([1, 'bbbb'])
with table_resource.open_reader() as reader:
    for rec in reader:
        print(rec)
```

1.11.4.7. Functions

This topic describes how to perform basic operations on functions by using PyODPS.

You can create user defined functions (UDFs) and use them in MaxCompute SQL.

Create a function

You can call the `create_function()` method of a MaxCompute entry object to create a function. Example:

```
# Reference a resource in the current project.
resource = o.get_resource('my_udf.py')
function = o.create_function('test_function', class_type='my_udf.Test', resources=[resource])
# Reference a resource in another project.
resource2 = o.get_resource('my_udf.py', project='another_project')
function2 = o.create_function('test_function2', class_type='my_udf.Test', resources=[resource2])
```

Delete a function

You can call the `delete_function()` method of a MaxCompute entry object to delete a function. You can also call the `drop()` method to delete a function. Example:

```
o.delete_function('test_function')
function.drop() # Call the drop() method if the function exists.
```

Update a function

You can call the `update()` method to update a function. Example:

```
function = o.get_function('test_function')
new_resource = o.get_resource('my_udf2.py')
function.class_type = 'my_udf2.Test'
function.resources = [new_resource, ]
function.update() # Update the function.
```

1.11.5. DataFrame

This topic describes how to create and manage a DataFrame object. This topic also provides information about how to use DataFrame to process data.

PyODPS provides an interface similar to pandas called PyODPS DataFrame. This interface operates on MaxCompute tables and makes full use of the capabilities of MaxCompute. You can also change the data source from MaxCompute tables to pandas DataFrame. This way, the same code can be executed on pandas.

For the complete DataFrame document, see [DataFrame](#).

DataFrame object operations

The following example demonstrates how to create a DataFrame object.

 **Note** The data used in the example is downloaded from [movielens 100K](#). You can download the data as needed.

Assume that the following tables exist: `pyodps_ml_100k_movies` that contains movie-related data, `pyodps_ml_100k_users` that contains user-related data, and `pyodps_ml_100k_ratings` that contains rating-related data.

1. If no MaxCompute object is provided in the runtime environment, you must create an object.

```
from odps import ODPS
o = ODPS(**your-access-id**, **your-secret-access-key**,
        project=**your-project**, endpoint=**your-end-point**)
```

2. You need only to specify a table object to create a DataFrame object.

```
from odps.df import DataFrame
users = DataFrame(o.get_table('pyodps_ml_100k_users'))
```

3. You can use the `dtypes` attribute to view the fields of the DataFrame object and the data types of the fields.

```
users.dtypes
odps.Schema {
  user_id      int64
  age          int64
  sex          string
  occupation   string
  zip_code     string
}
```

4. You can use the `head` method to have a quick preview on the first N data records.

```
users.head(10)
  user_id age sex  occupation zip_code
0     1  24  M  technician  85711
1     2  53  F    other  94043
2     3  23  M    writer  32067
3     4  24  M  technician  43537
4     5  33  F    other  15213
5     6  42  M  executive  98101
6     7  57  M administrator  91344
7     8  36  M administrator  05201
8     9  29  M    student  01002
9    10  53  M    lawyer  90703
```

5. If you do not want to view all the fields, you can perform the following operations as needed:
 - o Specify the fields that you want to query.

```
users[['user_id', 'age']].head(5)
  user_id age
0     1  24
1     2  53
2     3  23
3     4  24
4     5  33
```

- o Exclude several fields. Example:

```
users.exclude('zip_code', 'age').head(5)
  user_id sex occupation
0     1  M technician
1     2  F   other
2     3  M   writer
3     4  M technician
4     5  F   other
```

- Exclude some fields and add new fields based on computation. For example, add the `sex_bool` attribute and set it to `True` if sex is M. Otherwise, set it to `False`.

```
users.select(users.exclude('zip_code', 'sex'), sex_bool=users.sex == 'M').head(5)
  user_id age occupation sex_bool
0     1  24 technician   True
1     2  53   other   False
2     3  23   writer   True
3     4  24 technician   True
4     5  33   other   False
```

6. You can obtain the numbers of male and female users.

```
users.groupby(users.sex).agg(count=users.count())
  sex count
0  F   273
1  M   670
```

7. To divide users by occupation, you can obtain the first 10 occupations that have the largest population, and sort the occupations in descending order of population.

```
df = users.groupby('occupation').agg(count=users['occupation'].count())
df.sort(df['count'], ascending=False)[:10]
  occupation count
0  student   196
1  other   105
2  educator   95
3  administrator   79
4  engineer   67
5  programmer   66
6  librarian   51
7  writer   45
8  executive   32
9  scientist   31
```

Alternatively, you can use the `value_counts` method. The number of records returned by this method is limited by `options.df.odps.sort.limit`.

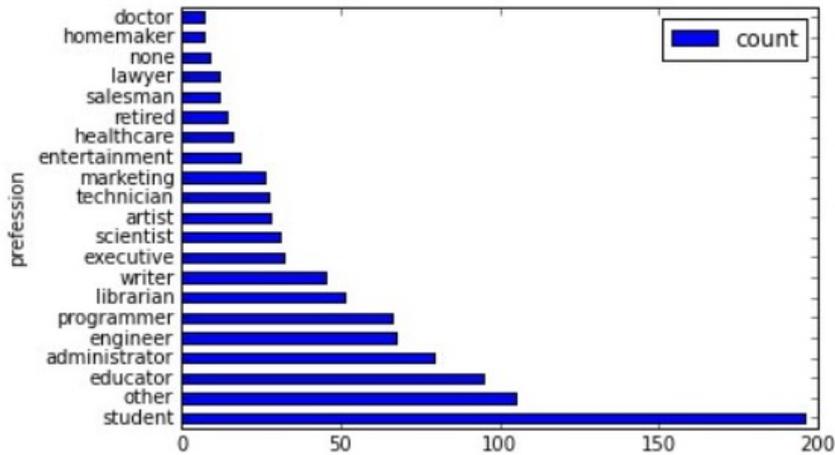
```
users.occupation.value_counts()[:10]
  occupation count
0  student   196
1  other   105
2  educator   95
3  administrator   79
4  engineer   67
5  programmer   66
6  librarian   51
7  writer   45
8  executive   32
9  scientist   31
```

8. Show data in a more intuitive graph.

```
%matplotlib inline
```

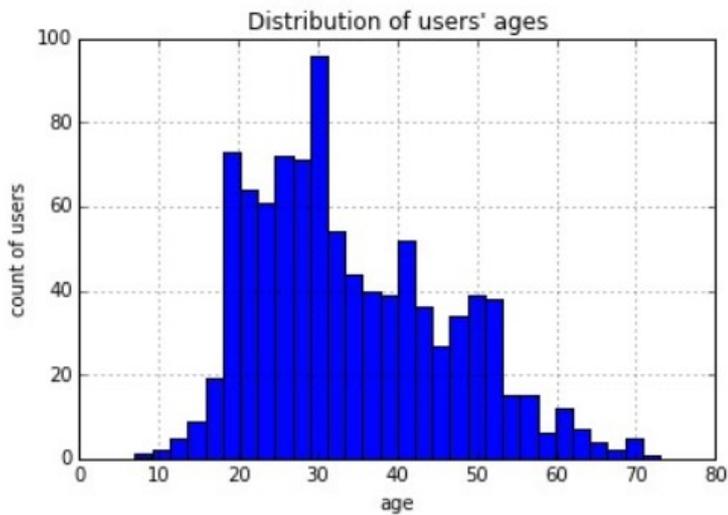
9. Use a horizontal column chart to visualize data.

```
users['occupation'].value_counts().plot(kind='barh', x='occupation', ylabel='profession')  
<matplotlib.axes._subplots.AxesSubplot at 0x10653cfd0>
```



10. Divide users into 30 groups by age and view the histogram of age distribution.

```
users.age.hist(bins=30, title="Distribution of users' ages", xlabel='age', ylabel='count of users')  
<matplotlib.axes._subplots.AxesSubplot at 0x10667a510>
```



11. Use `join` to join the three tables and save them as a new table.

```
movies = DataFrame(o.get_table('pyodps_ml_100k_movies'))
ratings = DataFrame(o.get_table('pyodps_ml_100k_ratings'))
o.delete_table('pyodps_ml_100k_lens', if_exists=True)
lens = movies.join(ratings).join(users).persist('pyodps_ml_100k_lens')
lens.dtypes
odps.Schema {
  movie_id          int64
  title             string
  release_date      string
  video_release_date string
  imdb_url          string
  user_id           int64
  rating            int64
  unix_timestamp    int64
  age               int64
  sex               string
  occupation        string
  zip_code          string
}
```

12. Divide users aged 0 to 80 into eight age groups.

```
labels = ['0-9', '10-19', '20-29', '30-39', '40-49', '50-59', '60-69', '70-79']
cut_lens = lens[lens, lens.age.cut(range(0, 81, 10), right=False, labels=labels).rename('age_group')]
```

13. View the first 10 data records of a single age in a group.

```
cut_lens['age_group', 'age'].distinct()[:10]
age_group age
0 0-9 7
1 10-19 10
2 10-19 11
3 10-19 13
4 10-19 14
5 10-19 15
6 10-19 16
7 10-19 17
8 10-19 18
9 10-19 19
```

14. View the total rating and average rating of users in each age group.

```
cut_lens.groupby('age_group').agg(cut_lens.rating.count().rename('total_rating'), cut_lens.rating.mean().rename('avg_rating'))
age_group avg_rating total_rating
0 0-9 3.767442 43
1 10-19 3.486126 8181
2 20-29 3.467333 39535
3 30-39 3.554444 25696
4 40-49 3.591772 15021
5 50-59 3.635800 8704
6 60-69 3.648875 2623
7 70-79 3.649746 197
```

DataFrame data processing

The following example demonstrates how to process DataFrame data.

 **Note** The data used in the example is downloaded from [Iris dataset](#). You can download the data as needed.

1. Create a test data table.

Use the table management feature of DataWorks to create a table, and then click **DDL mode**.

Enter the CREATE TABLE statement and submit the table. Example:

```
CREATE TABLE `pyodps_iris` (
  `sepallength` double COMMENT 'sepallength(cm)',
  `sepalwidth` double COMMENT 'sepalwidth(cm)',
  `petallength` double COMMENT 'petallength(cm)',
  `petalwidth` double COMMENT 'petalwidth(cm)',
  `name` string COMMENT 'name'
);
```

2. Upload test data.

Click the **Import** icon.

Enter the table name and upload the downloaded dataset.

Select **Match by location** and click **Import Data**.

3. Create a PyODPS node to store and run code.
4. Enter the code and click the Run icon. You can view the result in the **Runtime Log** section in the lower pane.

Code details:

```
from odps.df import DataFrame
from odps.df import output
iris = DataFrame(o.get_table('pyodps_iris')) # Create the DataFrame object iris from the MaxCompute table.
print iris.head(10)
print iris.sepallength.head(5) # Display part of the iris content.
# Use a user defined function to calculate the sum of two columns of iris.
print iris.apply(lambda row: row.sepallength + row.sepalwidth, axis=1, reduce=True, types='float').rename('sepaladd').head(3)
# Specify the output name and type of the function.
@output(['iris_add', 'iris_sub'], ['float', 'float'])
def handle(row):
# Use the yield keyword to return multiple rows of results.
    yield row.sepallength - row.sepalwidth, row.sepallength + row.sepalwidth
    yield row.petallength - row.petalwidth, row.petallength + row.petalwidth
# Display the results of the first five rows. axis=1 indicates that the axis of the column extends horizontally.
print iris.apply(handle, axis=1).head(5)
```

1.11.6. User experience enhancement

1.11.6.1. Command line

This topic describes command line enhancement.

PyODPS provides an enhanced command line tool. You can perform the following steps to configure and call the tool:

1. Import the PyODPS enhancement tool.

```
from odps.inter import setup, enter, teardown
```

2. Configure your account.

```
setup('**your-access-id**', '**your-access-key**', '**your-project**', endpoint='**your-endpoint**')
```

 Note

- If you do not specify the room parameter, the default room is used.
- After you configure an account, you do not need to enter the account information again.

3. Call the enter method to create a room object on a Python interactive interface.

```
room = enter()
o = room.odps
o.get_table('dual')
odps.Table
name: odps_test_sqltask_finance.`dual`
schema:
  c_int_a      : bigint
  c_int_b      : bigint
  c_double_a   : double
  c_double_b   : double
  c_string_a   : string
  c_string_b   : string
  c_bool_a     : boolean
  c_bool_b     : boolean
  c_datetime_a : datetime
  c_datetime_b : datetime
```

 Note The MaxCompute entry object is not automatically updated when you change the setup of the room. You must call `enter()` again to retrieve the new room object.

After you configure an account and call objects, you can store, retrieve, or delete objects in the room or delete the entire room.

- You can store commonly used MaxCompute tables or resources in the room. Example:

```
room.store('stored-table', o.get_table('dual'), desc='Simple stored table example')
```

- You can call the `display` method to display the stored objects as a table. Example:

```
room.display()
```

Result:

```
default name  desc
stored-table  Simple stored table example
iris          Iris dataset
```

- You can use `room['stored-table']` or `room.iris` to retrieve the stored objects. Example:

```
room['stored-table']
odps.Table
name: odps_test_sqltask_finance.`dual`
schema:
  c_int_a      : bigint
  c_int_b      : bigint
  c_double_a   : double
  c_double_b   : double
  c_string_a   : string
  c_string_b   : string
  c_bool_a     : boolean
  c_bool_b     : boolean
  c_datetime_a : datetime
  c_datetime_b : datetime
```

- You can call the `drop` method to delete objects in the room. Example:

```
room.drop('stored-table')
room.display()
default name  desc
iris         Iris dataset
```

- You can call the `teardown` method to delete a room. If no parameters are specified, the default room is deleted.

```
teardown()
```

1.11.6.2. IPython

This topic describes IPython enhancement.

PyODPS provides the IPython plug-in to facilitate MaxCompute operations.

IPython enhancement

Some commands are provided for command line enhancement.

- You can run the following code to load the plug-in:

```
%load_ext odps
%enter
```

The following result is returned:

```
<odps.inter.Room at 0x11341df10>
```

- In this case, the global `o` and `odps` variables can be retrieved. You can run the `o.get_table` or `odps.get_table` command to call a table.

```
o.get_table('dual')
odps.get_table('dual')
```

The following result is returned:

```
odps.Table
name: odps_test_sqltask_finance.`dual`
schema:
  c_int_a      : bigint
  c_int_b      : bigint
  c_double_a   : double
  c_double_b   : double
  c_string_a   : string
  c_string_b   : string
  c_bool_a     : boolean
  c_bool_b     : boolean
  c_datetime_a : datetime
  c_datetime_b : datetime
```

- You can run the following command to display the stored objects as a table:

```
%stores
```

The following result is returned:

```
default name desc
iris      Iris dataset
```

Object name completion

PyODPS enhances the code completion feature that is provided by IPython. When you write a statement such as `o.get_xxx`, the object name is automatically completed. In the following examples, `<tab>` is used to denote pressing the Tab key. When you enter the statement and encounter `<tab>`, press the Tab key.

- You can use the Tab key to complete the object name.

```
o.get_table(<tab>
```

- You can enter the first few characters of an object name and press the Tab key to complete it:

```
o.get_table('tabl<tab>
```

IPython auto-completes the table name that starts with tabl.

- This feature also completes the names of objects in different projects. Syntax:

```
o.get_table(project='project_name', name='tabl<tab>
o.get_table('tabl<tab>', project='project_name')
```

- If multiple matching objects exist, IPython provides a list. `options.completion_size` specifies the maximum number of objects in the list. The default value is 10.

SQL statements

PyODPS provides an SQL plug-in to execute MaxCompute SQL statements.

- You can use `%sql` to execute a single-line SQL statement. Example:

```
In [*]: %sql select * from pyodps_iris limit 5
|=====| 1 / 1 (100.00%) 3s
Out[*]:
  sepallength  sepalwidth  petallength  petalwidth  name
0     5.1     3.5     1.4     0.2  Iris-setosa
1     4.9     3.0     1.4     0.2  Iris-setosa
2     4.7     3.2     1.3     0.2  Iris-setosa
3     4.6     3.1     1.5     0.2  Iris-setosa
4     5.0     3.6     1.4     0.2  Iris-setosa
```

- You can use `%%sql` to execute a multiple-line SQL statement. Example:

```
In [*]: %%sql
.....: select * from pyodps_iris
.....: where sepallength < 5
.....: limit 5
.....:
|=====| 1 / 1 (100.00%) 15s
Out[*]:
  sepallength  sepalwidth  petallength  petalwidth  name
0     4.9     3.0     1.4     0.2  Iris-setosa
1     4.7     3.2     1.3     0.2  Iris-setosa
2     4.6     3.1     1.5     0.2  Iris-setosa
3     4.6     3.4     1.4     0.3  Iris-setosa
4     4.4     2.9     1.4     0.2  Iris-setosa
```

- To execute parameterized SQL statements, you can use `:parameter` to specify the parameter. Example:

```
In [1]: %load_ext odps
In [2]: mytable = 'dual'
In [3]: %sql select * from :mytable
|=====| 1 / 1 (100.00%) 2s
Out[3]:
  c_int_a  c_int_b  c_double_a  c_double_b  c_string_a  c_string_b  c_bool_a \
0     0     0    -1203     0     0    -1203  True
  c_bool_b  c_datetime_a  c_datetime_b
0  False  2012-03-30 23:59:58  2012-03-30 23:59:59
```

- For SQL runtime parameters, you can use `%set` to set a global parameter or use `SET` within an SQLCell to set a local parameter. The following example sets a local parameter, which does not affect the global setting.

```
In [*]: %%sql
        set odps.sql.mapper.split.size = 16;
        select * from pyodps_iris;
```

- The following example sets a global parameter. Global settings apply to all subsequent SQL statements.

```
In [*]: %set odps.sql.mapper.split.size = 16
```

Upload pandas DataFrame to MaxCompute tables

PyODPS provides commands to upload pandas DataFrame objects to MaxCompute tables.

You need only to use the `%persist` command. The first parameter `df` is the variable name. The second parameter `pyodps_pandas_df` is the MaxCompute table name.

```
import pandas as pd
import numpy as np
df = pd.DataFrame(np.arange(9).reshape(3, 3), columns=list('abc'))
%persist df pyodps_pandas_df
```

1.11.6.3. Jupyter Notebook

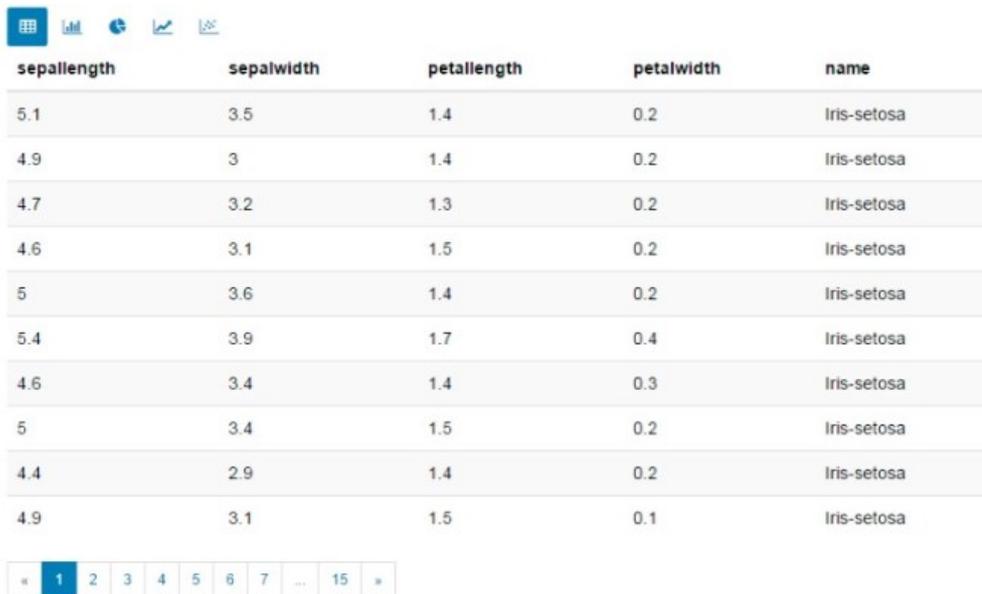
This topic describes Jupyter Notebook enhancement.

PyODPS enhances the result exploration and progress display features of Jupyter Notebook.

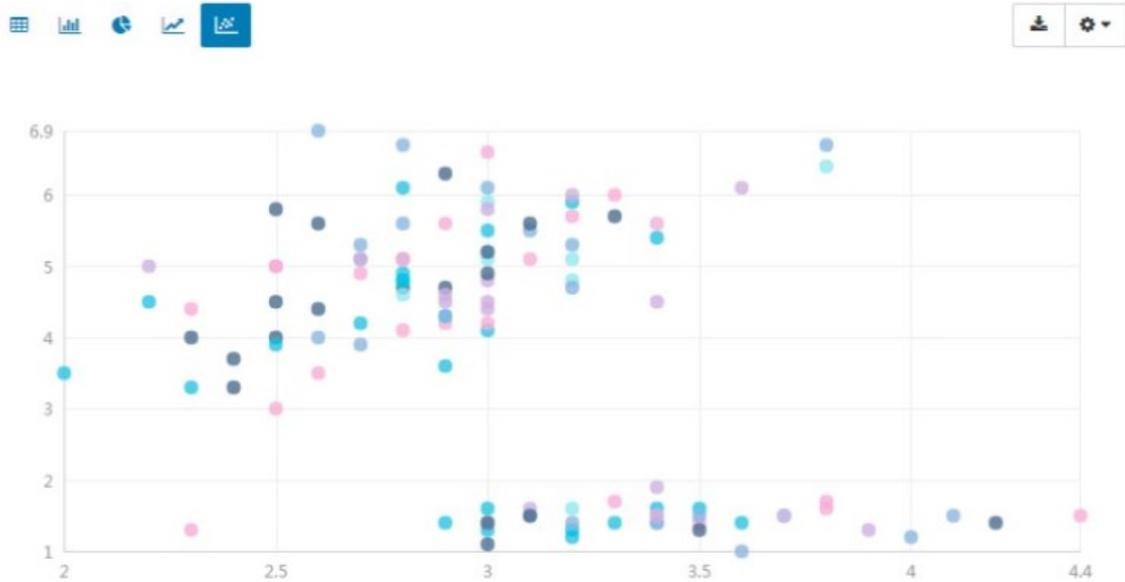
Result exploration

PyODPS provides a data exploration feature in Jupyter Notebook for SQLCell and DataFrame. You can use interactive data exploration tools to browse local data and create graphs.

1. If the execution result is a DataFrame object, PyODPS reads the result and displays it in a paged table. You can click a page number, the Previous button, or the Next button to browse the data.



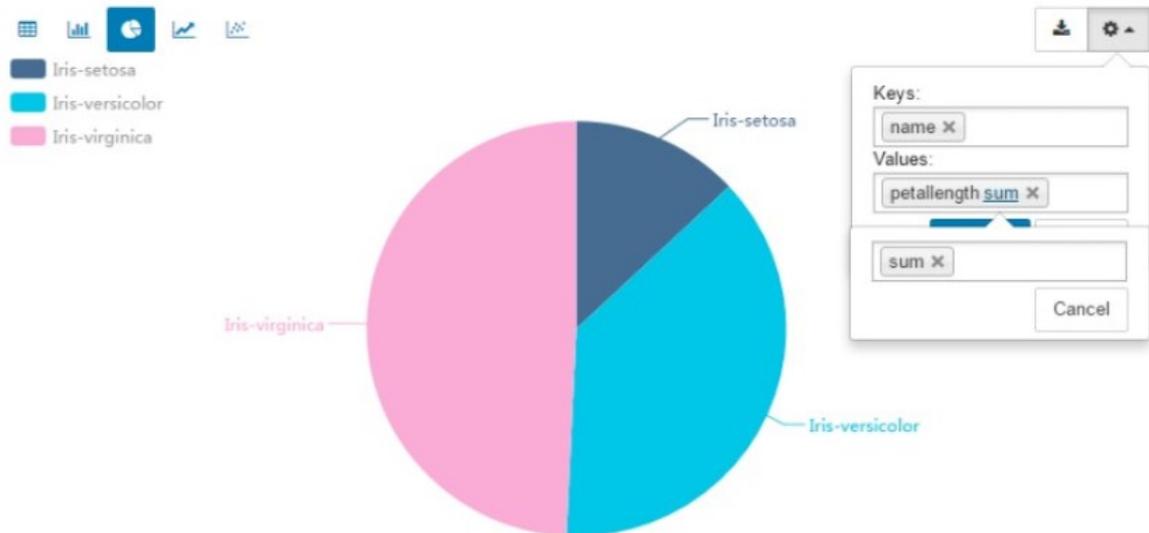
2. You can select other display modes on the top of the table to display the result in a column chart, pie chart, line chart, or scatter chart. The following figure shows a scatter chart created based on the default fields, which are the first three fields.



3. You can click the Settings icon in the upper-right corner of a graph to modify the settings. For example, set Groups to name, X Axis to petalength, and Y Axis to petalwidth. The result is shown in the following graph. The petalength-petalwidth settings display the data in a manner that is easy to understand.



For column charts and pie charts, you can select an aggregate function for the value fields. The default aggregate function for column charts is sum, and that for pie charts is count. You can click the function name next to the name of the value field to select another function. For line charts, the values on the x-axis cannot be null. If a value is null, the graph may not be correctly displayed.



4. After you make the graph, click the **Download** icon to save it.

Note To use this feature, you must install pandas and ipywidgets.

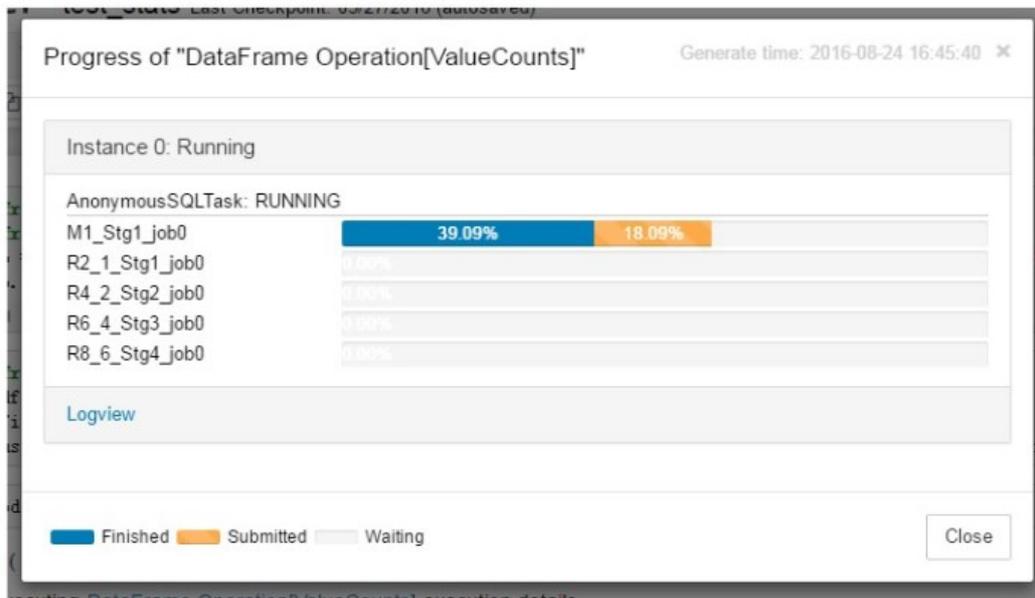
Progress display

The execution of large jobs takes extended periods of time. PyODPS provides progress bars to show the execution progress. When DataFrame jobs, machine learning jobs, or SQL statements that start with `%sql` are executed in Jupyter Notebook, a list of these jobs and their overall progress are displayed.

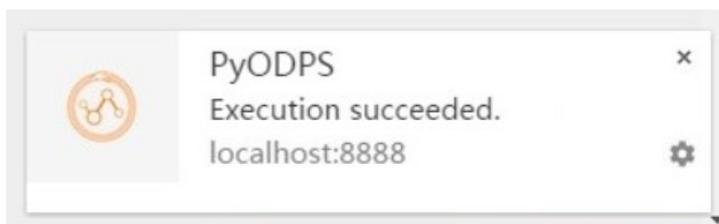
```
In [*]: from odps import ODPS
        from odps.df.examples import create_ionosphere
        o = ODPS(access_id, secret_access_key, project=project, endpoint=endpoint)
        df = create_ionosphere(o) ['a01', 'a02', 'a03', 'a04', 'class']
        df.calc_summary()
```



When you click a job name, a dialog box that displays the progress of each task in the job appears.



After the execution has been completed, a message that indicates whether the job is successful appears.



1.11.7. Configurations

This topic describes the configuration options provided by PyODPS.

PyODPS provides a series of configuration options. You can obtain them by using `odps.options`. Example:

```
from odps import options
# Set the lifecycle option to specify the lifecycle of all output tables.
options.lifecycle = 30
# Set the tunnel.string_as_binary option to True to use bytes instead of Unicode to download data of the STRING type.
options.tunnel.string_as_binary = True
# When you execute PyODPS DataFrames in MaxCompute, you can refer to the following configuration to set the limit to a relatively large value during a sort operation.
options.df.odps.sort.limit = 100000000
```

General configurations

Option	Description	Default value
end_point	The endpoint of MaxCompute.	None
default_project	The default project.	None
log_view_host	The hostname of Logview.	None

Option	Description	Default value
log_view_hours	The retention time of Logview. Unit: hours.	24
local_timezone	The time zone that is used. True indicates local time, and False indicates UTC. The time zone of pytz can also be used.	None
lifecycle	The lifecycle of all tables.	None
temp_lifecycle	The lifecycle of temporary tables.	1
biz_id	The user ID.	None
verbose	Specifies whether to display logs.	False
verbose_log	The log receiver.	None
chunk_size	The size of the write buffer.	1496
retry_times	The number of request retries.	4
pool_connections	The number of cached connections in the connection pool.	10
pool_maxsize	The maximum capacity of the connection pool.	10
connect_timeout	The connection timeout period.	5
read_timeout	The read timeout period.	120
api_proxy	The API proxy server.	None
data_proxy	The data proxy server.	None
completion_size	The limit on the number of object completion listing items.	10
notebook_repr_widget	Specifies whether to use interactive graphs.	True
sql.settings	Global hints for MaxCompute SQL.	None
sql.use_odps2_extension	Specifies whether to enable MaxCompute 2.0 language extension.	False

Data upload and download configurations

Option	Description	Default value
tunnel.endpoint	The endpoint of MaxCompute Tunnel.	None

Option	Description	Default value
tunnel.use_instance_tunnel	Specifies whether to use InstanceTunnel to obtain execution results.	True
tunnel.limit_instance_tunnel	Specifies whether to limit the number of data records obtained by using InstanceTunnel.	None
tunnel.string_as_binary	Specifies whether to use bytes instead of Unicode for data of the STRING type.	False

DataFrame configurations

Option	Description	Default value
interactive	Specifies whether DataFrames are used in an interactive environment.	Depending on the detection value
df.analyze	Specifies whether to enable functions that are not built in MaxCompute.	True
df.optimize	Specifies whether to enable full DataFrame optimization.	True
df.optimizes.pp	Specifies whether to enable DataFrame predicate pushdown optimization.	True
df.optimizes.cp	Specifies whether to enable DataFrame column pruning optimization.	True
df.optimizes.tunnel	Specifies whether to enable DataFrame tunnel optimization.	True
df.quote	Specifies whether to use a pair of grave accents (``) to mark field and table names in the backend of MaxCompute SQL.	True
df.libraries	The resource name of the third-party library that is used for DataFrame operations.	None
df.supersede_libraries	Specifies whether to use the self-uploaded NumPy to replace the version in the service.	False
df.odps.sort.limit	The default limit on the number of items that are added during a sort operation of DataFrames.	10000

Machine learning configurations

Option	Description	Default value
ml.xflow_settings	The XFlow execution configuration.	None
ml.xflow_project	The default XFlow project name.	algo_public
ml.use_model_transfer	Specifies whether to use ModelTransfer to obtain the Predictive Model Markup Language (PMML) files of models.	False
ml.model_volume	The name of the volume used by ModelTransfer.	pyodps_volume

1.11.8. API overview

This topic provides the links to PyODPS API documentation, which provides parameter descriptions and examples of each function:

- [Definitions](#)
- [DataFrame Reference](#)

1.11.9. FAQ

This topic provides the best practices and some frequently asked questions about PyODPS, which help you efficiently develop PyODPS programs.

How do I view the current PyODPS version?

Run the following commands:

```
import odps
print(odps.__version__)
```

How do I troubleshoot the "Project not found" error?

This problem is generally caused by invalid endpoint configurations. You must check the configurations and rectify the problem. You also need to check whether the position of the MaxCompute object parameter is correct.

How do I manually specify a Tunnel endpoint?

You can create your MaxCompute object with the tunnel_endpoint parameter specified, as shown in the following code. Replace the content enclosed with asterisks (*) with actual parameter values and remove the asterisks (*).

```
from odps import ODPS
o = ODPS(**your-access-id**, **your-secret-access-key**, **your-default-project**,
        endpoint=**your-end-point**, tunnel_endpoint=**your-tunnel-endpoint**)
```

How do I troubleshoot the "project is protected" error reported while data is being read?

The project security policy does not allow you to read data from tables. To retrieve all the data, you can use the following solutions:

- Contact the project owner to add exception rules.
- Use DataWorks or other masking tools to mask the data and then export the data to an unprotected project before reading it.

To retrieve part of the data, you can use the following solutions:

- Use `o.execute_sql('select * from <table_name>').open_reader()` .
- Use `o.get_table('<table_name>').to_df()` in DataFrame.

I can only retrieve a maximum of 10,000 data records by executing SQL command `open_reader`. How do I retrieve more than 10,000 data records?

Use `create table as select ...` to save the SQL result to a table, and then use `table.open_reader` to read data.

How do I troubleshoot the "ODPSError: ODPS entrance should be provided" error reported when I upload pandas DataFrame to MaxCompute?

This error is reported because the global MaxCompute object cannot be found. Use one of the following methods to resolve this problem:

- If you use the room mechanism `%enter` , configure the global MaxCompute object.
- Call the `to_global` method for the MaxCompute object.
- Use the following MaxCompute parameter:

```
DataFrame(pd_df).persist('your_table', odps=odps)
```

How do I use `max_pt` in DataFrame?

Use the `odps.df.func` module to call built-in functions of MaxCompute. Example:

```
from odps.df import func
df = o.get_table('your_table').to_df()
df[df.ds == func.max_pt('your_project.your_table')] # ds is a partition column.
```

How do I troubleshoot the "table lifecycle is not specified in mandatory mode" error reported when I use DataFrame to write data to a table?

Your project requires that every table be created with a lifecycle. Therefore, you must make the following configuration every time you run your own code:

```
from odps import options
options.lifecycle = 7 # You can also set this parameter to your expected lifecycle in days.
```

How do I traverse each row of data in PyODPS DataFrame?

PyODPS DataFrame does not support this feature. PyODPS DataFrame focuses on handling large volumes of data. Traversing data is inefficient.

We recommend that you use the `apply` or `map_reduce` method of DataFrame to parallelize your serial traverse operations.

If you confirm that data traversing is necessary in your scenario and that the cost is acceptable, you can use the `to_pandas` method to convert your DataFrame to pandas DataFrame. You can also store DataFrame as a table and use `read_table` or Tunnel to read data.

1.12. Java sandbox limits

This topic describes the limits imposed by Java sandboxes on MaxCompute MapReduce and user-defined function (UDF) programs in distributed environments.

 **Note** The main programs of MapReduce jobs are not subject to these limits.

Limits:

- Direct access to local files is not allowed. You can access files only by using interfaces provided by MaxCompute MapReduce or MaxCompute Graph.
- Direct access to distributed file systems is not allowed. You can use only MaxCompute MapReduce or MaxCompute Graph to access tables.
- Java Native Interface (JNI) calls are not allowed.
- Java threads cannot be created, and Linux commands cannot be executed by sub-threads.
- Network access operations such as acquiring local IP addresses are not allowed.
- Java reflection limit: The `suppressAccessChecks` permission is prohibited. You cannot set a private attribute or method accessible to read private attributes or call private methods.

An access denied error is returned when you perform one of the preceding operations.

1.13. Volume lifecycle management

1.13.1. Overview

This topic provides an overview of the lifecycle of a MaxCompute volume.

In the previous versions of MaxCompute, a partition in a volume does not have a lifecycle and can exist indefinitely. Users must manage the lifecycles of volumes. In some cases, user management of volume lifecycles may cause a problem. For example, if the account used to delete a partition is different from the account used to create the partition, the delete operation fails. The new feature of volume lifecycle management in the current version solves this problem.

For more information about the volume lifecycle management feature, see the subsequent topic *Volume lifecycle operations*.

1.13.2. Manage the lifecycle of a volume

This topic describes how to manage the lifecycle of a MaxCompute volume.

Create a volume with a specified lifecycle

Example:

```
odps@ your_project>fs -mkv test_volume -lifecycle 7 "this is a test volume";
OK
```

Modify the lifecycle of a volume

Example:

```
odps@ your_project>fs -alter test_volume -lifecycle 3;
OK
```

View the lifecycle of a volume

Example:

```
odps@ your_project>fs -meta test_volume;  
Comment: "this is a test volume"  
Length: 0  
File number: 0  
Lifecycle: 3  
OK
```

1.14. Spark on MaxCompute

1.14.1. Overview

This topic describes the definition and core features of Spark on MaxCompute.

Spark on MaxCompute is a solution developed by Alibaba Cloud to enable the seamless use of Spark on the MaxCompute platform. It supplements a wide variety of features to MaxCompute.

Spark on MaxCompute ensures the same user experience as native Spark and offers native Spark components and APIs. Spark on MaxCompute can access MaxCompute data sources and enhance security for multi-tenant scenarios. Spark on MaxCompute can also act as a management platform to share resources, storage, and user systems between Spark and MaxCompute jobs and ensure high performance at low costs. Spark can work with MaxCompute to create more efficient data processing solutions. Spark community applications can run seamlessly on Spark on MaxCompute.

Spark on MaxCompute has an independent data development node in DataWorks and supports data development in DataWorks.

1.14.2. Project resources

This topic describes the project resources that you must obtain before you can use Spark on MaxCompute.

Before you use Spark on MaxCompute, you must obtain the following project resources:

- Spark on MaxCompute release package: You must download the latest [Spark on MaxCompute](#) release package.
- Spark on MaxCompute plug-in: This is an open-source plug-in. You can download it from [Aliyun Cupid SDK](#).

After you prepare the preceding project resources, you must complete environment configuration. Then, you can run related GitHub demos.

1.14.3. Environment settings

1.14.3.1. Decompress the Spark on MaxCompute release package

This topic describes the structure of the obtained folders and files after you decompress the Spark on MaxCompute release package.

Decompress the downloaded Spark on MaxCompute release package. The following code shows the structure of the obtained folders and files:

```
.
|-- R
|-- RELEASE
|-- __spark_libs__.zip
|-- bin
|-- conf
|-- cupid
|-- derby.log
|-- examples
|-- jars
|-- logs
|-- metastore_db
|-- python
|-- sbin
|-- yarn
```

1.14.3.2. Set environment variables

This topic describes how to set the environment variables required by Spark on MaxCompute.

Set environment variables based on your business requirements. The main environment variables are JAVA_HOME and SPARK_HOME.

Set JAVA_HOME

```
export JAVA_HOME=/path/to/jdk
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$JAVA_HOME/bin:$PATH
```

Set SPARK_HOME

```
export SPARK_HOME=/path/to/spark_extracted_package
export PATH=$SPARK_HOME/bin:$PATH
```

If you use SparkR, install R in the `/home/admin/R` directory. Then, run the following command to set the path:

```
export PATH=/home/admin/R/bin/:$PATH
```

If you use PySpark, install Python 2.7. Then, run the following command to set the path:

```
export PATH=/path/to/python/bin/:$PATH
```

1.14.3.3. Configure Spark-defaults.conf

This topic describes how to configure the Spark-defaults.conf file, which is required when you use Spark on MaxCompute.

The `$SPARK_HOME/conf` directory contains a file named Spark-defaults.conf. Before you submit a Spark task to MaxCompute, you must configure your MaxCompute account in this file.

The following information is the default configuration in the file. You need only to specify the blank parameters based on your account information.

```
# OdpsAccount Info Setting
spark.hadoop.odps.project.name=
spark.hadoop.odps.access.id=
spark.hadoop.odps.access.key=
spark.hadoop.odps.end.point=
#spark.hadoop.odps.moye.trackurl.host=
#spark.hadoop.odps.cupid.webproxy.endpoint=
spark.sql.catalogImplementation=odps
# Spark-shell Setting
spark.driver.extraJavaOptions -Dscala.repl.reader=com.aliyun.odps.spark_repl.OdpsInteractiveReader -Dscala.usejavacp=true
# SparkR Setting
# odps.cupid.spark.r.archive=/path/to/R-PreCompile-Package.zip
# Cupid Longtime Job
# spark.hadoop.odps.cupid.engine.running.type=longtime
# spark.hadoop.odps.cupid.job.capability.duration.hours=8640
# spark.hadoop.odps.moye.trackurl.dutation=8640
# spark.r.command=/home/admin/R/bin/Rscript
# spark.hadoop.odps.cupid.disk.driver.enable=false
spark.hadoop.odps.cupid.bearer.token.enable=false
spark.hadoop.odps.exec.dynamic.partition.mode=nonstrict
```

1.14.4. Quick start

This topic describes the basic operations for you to get started with Spark on MaxCompute.

1. Obtain and decompress the [Spark on MaxCompute](#) release package.
2. Set environment variables.

```
export SPARK_HOME=/path/to/spark-2.1.0-private-cloud-v3.1.0
export JAVA_HOME=/path/to/java/
```

3. Configure the spark-defaults.conf file.

```
cp $SPARK_HOME/conf/spark-defaults.conf.template $SPARK_HOME/conf/spark-defaults.conf
```

Specify the blank parameters in \$SPARK_HOME/conf/spark-defaults.conf.

```
# OdpsAccount Info Setting
spark.hadoop.odps.project.name=
spark.hadoop.odps.access.id=
spark.hadoop.odps.access.key=
spark.hadoop.odps.end.point=
#spark.hadoop.odps.moye.trackurl.host=
#spark.hadoop.odps.cupid.webproxy.endpoint=
spark.sql.catalogImplementation=odps
# spark-shell Setting
spark.driver.extraJavaOptions -Dscala.repl.reader=com.aliyun.odps.spark_repl.OdpsInteractiveReader -Dscala.usejavacp=true
# SparkR Setting
# odps.cupid.spark.r.archive=/path/to/R-PreCompile-Package.zip
# Cupid Longtime Job
# spark.hadoop.odps.cupid.engine.running.type=longtime
# spark.hadoop.odps.cupid.job.capability.duration.hours=8640
# spark.hadoop.odps.moye.trackurl.dutation=8640
# spark.r.command=/home/admin/R/bin/Rscript
# spark.hadoop.odps.cupid.disk.driver.enable=false
spark.hadoop.odps.cupid.bearer.token.enable=false
spark.hadoop.odps.exec.dynamic.partition.mode=nonstrict
```

4. Prepare spark-example.

```
git clone https://github.com/aliyun/aliyun-cupid-sdk.git
cd aliyun-cupid-sdk
mvn -T 1C clean install -DskipTests
```

5. Run SparkPi.

```
cd $SPARK_HOME
bin/spark-submit --master yarn-cluster --class com.aliyun.odps.spark.examples.SparkPi /path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

If the following output is displayed, your operation is successful. Other logs may be included in the output.

```
18/02/09 15:52:28 INFO Client: Application report for application_1518162700322_1635034099 (state: FINISHED)
18/02/09 15:52:28 INFO Client:
client token: N/A
diagnostics: N/A
ApplicationMaster host: ***
ApplicationMaster RPC port: ***
queue: queue
start time: 1518162732343
final status: SUCCEEDED
tracking URL: http://***:80/proxyview/jobview/?h=http://***:80/api&p=odps_smoke_test&i=20180209075148695gbkp8e01&t=spark&id=application_1518162700322_1635034099&metaname=20180209075148695gbkp8e01&token=YkhjNXJWZ0dvdzVScXFVQWpCQWMra1RSZHVFPsXPRFBT09CTzoxMzY1OTM3MTUwNzcyMjEzLDE1MTg0MjE5MzUseyJTdGF0ZW1lbnQiOlt7IkFjdGlvbiil6WyJvZHBzOlJlYXQixSwiRWZmZWN0ljoIQWxsY3ciLCJSZXNvdXJzSI6WyJhY3M6b2Rwcz0qOnByb2plY3RzL29kcHNfc21va2VfdGVzdC9pbmN0YW55ZXMvMjAxODAyMDkwNzUxNDg2OTVnYmtwOGUwMSJdfV0silZlcnNpb24iOillxln0=
user: user
18/02/09 15:52:28 INFO ShutdownHookManager: Shutdown hook called
18/02/09 15:52:28 INFO ShutdownHookManager: Deleting directory /tmp/spark-d77416ad-79a8-49f7-931d-0533663b5d85
```

1.14.5. Demo

This topic provides a demo on how to use Spark on MaxCompute.

The demo consists of the following steps:

1. Modify the configuration file.

Decompress the Spark package, go to the `conf` directory, and modify the following configuration items in the configuration file `spark-defaults.conf`:

```
spark.hadoop.odps.project.name=xxxx
spark.hadoop.odps.access.id=xxxx
spark.hadoop.odps.access.key=xxxx
spark.hadoop.odps.end.point=http://service.xxxx.xxxx.xxxx.qd-inc.com:80/api
spark.hadoop.odps.cupid.distributedcache.mincopy=3
spark.hadoop.odps.cupid.distributedcache.maxcopy=3
spark.hadoop.odps.cupid.proxy.domain.name=jobview.xxxx.xxxx.xxxx.com
spark.hadoop.odps.cupid.history.server.address=10.xx.xx.xx:18080
```

Note

- You can obtain the first four configuration items from the `web_component.conf` file in the `/cloud/app/odps-service-console/CupidFrontendServer#/cupid_web_proxy/current/conf/local/` directory.
- `spark.hadoop.odps.cupid.proxy.domain.name`: specifies the domain name of `odps_jobview_server_dns`. Remove `sparkui` at the beginning of the domain name.
- `spark.hadoop.odps.cupid.history.server.address`: specifies the IP address of the AG machine. Add the port number 18080 to the end of the IP address.

2. Start the program.

Go to the directory where the Spark package is decompressed and run the following commands:

```
bin/spark-submit
--class com.aliyun.odps.spark.examples.WordCount --master yarn-cluster
examples/spark-examples-2.0.0-SNAPSHOT-shaded.jar
```

3. Access the link in the command output.

Logview

The screenshot shows the MaxCompute Logview interface. At the top, there is a table for 'ODPS Instance' with columns: URL, Project, InstanceID, Owner, StartTime, EndTime, Latency, Status, Priority, SourceURL, and Tool. Below this is a diagram showing a 'CUPID' node connected to a 'cupid_task' node. At the bottom, there is a table for 'ODPS Tasks' with columns: Name, Type, Status, Result, Detail, History, StartTime, EndTime, Latency, and TimeLine.

URL	Project	InstanceID	Owner	StartTime	EndTime	Latency	Status	Priority	SourceURL	Tool
http://service...	odps_who...	2019024033...	ALYUAppsc000...	24/09/2019, 11:33:00	-	00:03:08	Running	1	http://...	No Link (not internal project)

Name	Type	Status	Result	Detail	History	StartTime	EndTime	Latency	TimeLine
cupid_task	CUPID	Running				24/09/2019, 11:33:00	-	00:03:08	

SparkUI

Spark Jobs (7)

User: admin
 Total Uptime: 18 s
 Scheduling Mode: FIFO
 Completed Jobs: 3

Event Timeline

Completed Jobs (3)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
2	take at WordCount.scala:31	2019/09/24 11:33:14	41 ms	1/1 (1 skipped)	5/5 (10 skipped)
1	take at WordCount.scala:31	2019/09/24 11:33:14	77 ms	1/1 (1 skipped)	4/4 (10 skipped)
0	take at WordCount.scala:31	2019/09/24 11:33:13	0.8 s	2/2	1 (1)

Executors

Summary

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write
Active(2)	0	0.0 B / 1.2 GB	0.0 B	2	0	0	20	20	1 s (47 ms)	0.0 B	2.5 KB	4.6 KB
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B
Total(2)	0	0.0 B / 1.2 GB	0.0 B	2	0	0	20	20	1 s (47 ms)	0.0 B	2.5 KB	4.6 KB

Executors

Show: 20 entries

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Thread	Dump
driver	10.20.0.36:53360	Active	0	0.0 B / 384.1 MB	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B	stdout	Thread stderr Dump
1	456a09211.cloud.a11.amtest43.24553	Active	0	0.0 B / 384.1 MB	0.0 B	1	0	0	11	11	0.6 s (24 ms)	0.0 B	596 B	3.2 KB	stdout	Thread stderr Dump
2	456a09214.cloud.a11.amtest43.24557	Active	0	0.0 B / 384.1 MB	0.0 B	1	0	0	9	9	0.7 s (23 ms)	0.0 B	1.9 KB	1.4 KB	stdout	Thread stderr Dump

Showing 1 to 3 of 3 entries

Spark History

Spark Jobs (7)

User: admin
 Total Uptime:
 Scheduling Mode: FIFO
 Completed Jobs: 3

Event Timeline

Completed Jobs (3)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
2	take at WordCount.scala:31	2019/09/24 11:54:22	65 ms	1/1 (1 skipped)	5/5 (10 skipped)
1	take at WordCount.scala:31	2019/09/24 11:54:22	57 ms	1/1 (1 skipped)	4/4 (10 skipped)
0	take at WordCount.scala:31	2019/09/24 11:54:21	0.9 s	2/2	1 (1)

1.14.6. Common cases

1.14.6.1. WordCount example

This topic provides an example of using Spark on MaxCompute to run WordCount.

You must download the GitHub project and compile the project before you can run related demos.

```
git clone https://github.com/aliyun/aliyun-cupid-sdk.git
-- Download a GitHub project.
cd aliyun-cupid-sdk
mvn -T 1C clean install -DskipTests
-- Compile the GitHub project.
```

After you complete the preceding steps, JAR packages are generated. These JAR packages will be used to run the demos in this and the subsequent topics.

Example:

```
package com.aliyun.odps.spark.examples
import org.apache.spark.sql.SparkSession
object WordCount {
  def main(args: Array[String]) {
    val spark = SparkSession
      .builder()
      .appName("WordCount")
      .getOrCreate()
    val sc = spark.sparkContext
    try {
      sc.parallelize(1 to 100, 10).map(word => (word, 1)).reduceByKey(_ + _, 10).take(100).foreach(println)
    } finally {
      sc.stop()
    }
  }
}
```

Run the following command to submit the job:

```
bin/spark-submit \
--master yarn-cluster \
--class com.aliyun.odps.spark.examples.WordCount \
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

1.14.6.2. OSS access example

This topic provides an example of using Spark on MaxCompute to access OSS.

Example:

```
package com.aliyun.odps.spark.examples.oss
import org.apache.spark.sql.SparkSession
object SparkUnstructuredDataCompute {
  def main(args: Array[String]) {
    val spark = SparkSession
      .builder()
      .appName("SparkUnstructuredDataCompute")
      .config("spark.hadoop.fs.oss.accessKeyId", "****")
      .config("spark.hadoop.fs.oss.accessKeySecret", "****")
      .config("spark.hadoop.fs.oss.endpoint", "oss-cn-hangzhou-zmf.aliyuncs.com")
      .getOrCreate()
    val sc = spark.sparkContext
    try {
      val pathIn = "oss://bucket/inputdata/"
      val inputData = sc.textFile(pathIn, 5)
      val cnt = inputData.count
      println(s"count: $cnt")
    } finally {
      sc.stop()
    }
  }
}
```

Run the following command to submit the job:

```
./bin/spark-submit
--jars cupid/hadoop-aliyun-package-3.0.0-alpha2-odps-jar-with-dependencies.jar
--class com.aliyun.odps.spark.examples.oss.SparkUnstructuredDataCompute
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

1.14.6.3. MaxCompute table read and write example

This topic provides an example of using Spark on MaxCompute to read data from and write data to a MaxCompute table and convert data to a Spark resilient distributed dataset (RDD).

 **Notice** The project and table specified in the demo must exist or be changed to the specific project and table.

Example:

```
package com.aliyun.odps.spark.examples
import com.aliyun.odps.data.Record
import com.aliyun.odps.{ PartitionSpec, TableSchema}
import org.apache.spark.odps.OdpsOps
import org.apache.spark.sql.SparkSession
import scala.util.Random
object OdpsTableReadWrite {
  def main(args: Array[String]) {
    val spark = SparkSession
      .builder()
      .appName("OdpsTableReadWrite")
      .getOrCreate()
    val sc = spark.sparkContext
    val projectName = sc.getConf.get("odps.project.name")
    try {
```

```

val odpsOps = new OdpsOps(sc)
// read from normal table via rdd api
val rdd_0 = odpsOps.readTable(
  projectName,
  "cupid_wordcount",
  (r: Record, schema: TableSchema) => (r.getString(0), r.getString(1))
)
//read from single partition column table via rdd api
val rdd_1 = odpsOps.readTable(
  projectName,
  "dfctest_single_parted",
  Array("pt=20160101"),
  (r: Record, schema: TableSchema) => (r.getString(0), r.getString(1), r.getString("pt"))
)
// read from multi partition column table via rdd api
val rdd_2 = odpsOps.readTable(
  projectName,
  "dfctest_parted",
  Array("pt=20160101,hour=12"),
  (r: Record, schema: TableSchema) => (r.getString(0), r.getString(1), r.getString("pt"), r.getString(3))
)
// read with multi partitionSpec definition via rdd api
val rdd_3 = odpsOps.readTable(
  projectName,
  "cupid_partition_table1",
  Array("pt1=part1,pt2=part1", "pt1=part1,pt2=part2", "pt1=part2,pt2=part3"),
  (r: Record, schema: TableSchema) => (r.getString(0), r.getString(1), r.getString("pt1"), r.getString("pt2"))
)
// save rdd into normal table
val transfer_0 = (v: Tuple2[String, String], record: Record, schema: TableSchema) => {
  record.set("id", v._1)
  record.set(1, v._2)
}
odpsOps.saveToTable(projectName, "cupid_wordcount_empty", rdd_0, transfer_0, true)
// save rdd into partition table with single partition spec
val transfer_1 = (v: Tuple2[String, String], record: Record, schema: TableSchema) => {
  record.set("id", v._1)
  record.set("value", v._2)
}
odpsOps.saveToTable(projectName, "cupid_partition_table1", "pt1=test,pt2=dev", rdd_0, transfer_1, true)
// dynamic save rdd into partition table with multiple partition spec
val transfer_2 = (v: Tuple2[String, String], record: Record, part: PartitionSpec, schema: TableSchema) => {
  record.set("id", v._1)
  record.set("value", v._2)
  val pt1_value = if (new Random().nextInt(10) % 2 == 0) "even" else "odd"
  val pt2_value = if (new Random().nextInt(10) % 2 == 0) "even" else "odd"
  part.set("pt1", pt1_value)
  part.set("pt2", pt2_value)
}
odpsOps.saveToTableForMultiPartition(projectName, "cupid_partition_table1", rdd_0, transfer_2, true)
} catch {
  case ex: Exception => {
    throw ex
  }
} finally {
  sc.stop
}

```

```
}  
}
```

Run the following command to submit the job:

```
bin/spark-submit \  
--master yarn-cluster \  
--class com.aliyun.odps.spark.examples.OdpsTableReadWrite \  
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

You can use one of the following methods to adjust the MaxCompute table read concurrency:

- Change the value of the `spark.hadoop.odps.input.split.size` parameter. A larger value indicates fewer Map tasks. The default value is 256 MB.
- Set `numPartitions` in `OdpsOps.readTable`. The value determines the number of Map tasks. The number is calculated based on `spark.hadoop.odps.input.split.size`.

1.14.6.4. MaxCompute Table Spark-SQL example

This topic provides an example of using `SQLContext` in Spark on MaxCompute to read data from and write data to a MaxCompute Table.

Notice

- The project and table specified in the demo must exist or be changed to the specific project and table.
- `Spark-defaults.conf` must contain the setting `spark.sql.catalogImplementation = odps`.

Example:

```

package com.aliyun.odps.spark.examples
import org.apache.spark.sql.SparkSession
object OdpsTableReadWriteViaSQL {
  def main(args: Array[String]) {
    // please make sure spark.sql.catalogImplementation=odps in spark-defaults.conf
    // to enable odps catalog
    val spark = SparkSession
      .builder()
      .appName("OdpsTableReadWriteViaSQL")
      .getOrCreate()
    val projectName = spark.sparkContext.getConf.get("odps.project.name")
    val tableName = "cupid_wordcount"
    // get a ODPS table as a DataFrame
    val df = spark.table(tableName)
    println(s"df.count: ${df.count()}")
    // Just do some query
    spark.sql(s"select * from $tableName limit 10").show(10, 200)
    spark.sql(s"select id, count(id) from $tableName group by id").show(10, 200)
    // any table exists under project could be use
    // productRevenue
    spark.sql(
      """
      |SELECT product,
      |  category,
      |  revenue
      |FROM
      | (SELECT product,
      |  category,
      |  revenue,
      |  dense_rank() OVER (PARTITION BY category
      |    ORDER BY revenue DESC) AS rank
      | FROM productRevenue) tmp
      |WHERE rank <= 2
      """
    ).stripMargin).show(10, 200)
    spark.stop()
  }
}

```

Run the following command to submit the job:

```

bin/spark-submit \
--master yarn-cluster \
--class com.aliyun.odps.spark.examples.OdpsTableReadWrite \
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar

```

1.14.6.5. Example of the self-developed client mode

This topic provides an example of the self-developed client mode in Spark on MaxCompute.

For security purposes, machines in MaxCompute cannot be directly connected. Therefore, the yarn-client mode in native Spark cannot be used. To enable interaction, the MaxCompute team developed a proprietary client mode.

Example:

```
package com.aliyun.odps.spark.examples
import com.aliyun.odps.cupid.client.spark.client.CupidSparkClientRunner
object SparkClientNormalFT {
  def main(args: Array[String]) {
    val cupidSparkClient = CupidSparkClientRunner.getReadyCupidSparkClient()
    val jarPath = args(0) //client-jobexamples jar path
    val sparkClientNormalApp = new SparkClientNormalApp(cupidSparkClient)
    sparkClientNormalApp.runNormalJob(jarPath)
    cupidSparkClient.stopRemoteDriver()
  }
}
```

Run the following command to submit the job:

```
java -cp \
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar:$SPARK_HOME/jars/* \
com.aliyun.odps.spark.examples.SparkClientNormalFT /path/to/aliyun-cupid-sdk/examples/client-jobexamples/target/client-jobexamples_2.11-1.0.0-SNAPSHOT.jar
```

1.14.6.6. MaxCompute Table PySpark example

This topic provides an example of using PySpark in Spark on MaxCompute to read data from and write data to a MaxCompute table.

Example:

```
from odps.odps_sdk import OdpsOps
from pyspark import SparkContext, SparkConf
from pyspark.sql import SQLContext, DataFrame
if __name__ == '__main__':
    conf = SparkConf().setAppName("odps_pyspark")
    sc = SparkContext(conf=conf)
    sql_context = SQLContext(sc)
    project_name = "cupid_testa1"
    in_table_name = "cupid_wordcount"
    out_table_name = "cupid_wordcount_py"
    normal_df = OdpsOps.read_odps_table(sql_context, project_name, in_table_name)
    for i in normal_df.sample(False, 0.01).collect():
        print i
    print "Read normal odps table finished"
    OdpsOps.write_odps_table(sql_context, normal_df.sample(False, 0.001), project_name, out_table_name)
    print "Write normal odps table finished"
```

Run the following command to submit the job:

```
spark-submit \
--master yarn-cluster \
--jars /path/to/aliyun-cupid-sdk/external/cupid-datasource/target/cupid-datasource_2.11-1.0.0-SNAPSHOT.jar \
--py-files /path/to/aliyun-cupid-sdk/examples/spark-examples/src/main/python/odps.zip \
/path/to/aliyun-cupid-sdk/examples/spark-examples/src/main/python/odps_table_rw.py
```

1.14.6.7. MLlib example

This topic provides an example of using MLlib in Spark on MaxCompute.

We recommend that you use OSS for read and write operations in the MLlib model.

Example:

```
package com.aliyun.odps.spark.examples.mllib
import org.apache.spark.mllib.clustering.KMeans._
import org.apache.spark.mllib.clustering.{ KMeans, KMeansModel}
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.sql.SparkSession
object KmeansModelSaveToOss {
  val modelOssDir = "oss://bucket/kmeans-model"
  def main(args: Array[String]) {
    //1. train and save the model
    val spark = SparkSession
      .builder()
      .appName("KmeansModelSaveToOss")
      .config("spark.hadoop.fs.oss.accessKeyId", "****")
      .config("spark.hadoop.fs.oss.accessKeySecret", "****")
      .config("spark.hadoop.fs.oss.endpoint", "****")
      .getOrCreate()
    val sc = spark.sparkContext
    val points = Seq(
      Vectors.dense(0.0, 0.0),
      Vectors.dense(0.0, 0.1),
      Vectors.dense(0.1, 0.0),
      Vectors.dense(9.0, 0.0),
      Vectors.dense(9.0, 0.2),
      Vectors.dense(9.2, 0.0)
    )
    val rdd = sc.parallelize(points, 3)
    val initMode = K_MEANS_PARALLEL
    val model = KMeans.train(rdd, k = 2, maxIterations = 2, runs = 1, initMode)
    val predictResult1 = rdd.map(feature => "cluster id: " + model.predict(feature) + " feature:" + feature.toArray.mkString(",")).collect
    println("modelOssDir=" + modelOssDir)
    model.save(sc, modelOssDir)
    //2. predict from the oss model
    val modelLoadOss = KMeansModel.load(sc, modelOssDir)
    val predictResult2 = rdd.map(feature => "cluster id: " + modelLoadOss.predict(feature) + " feature:" + feature.toArray.mkString(",")).collect
    assert(predictResult1.size == predictResult2.size)
    predictResult2.foreach(result2 => assert(predictResult1.contains(result2)))
  }
}
```

Run the following command to submit the job:

```
./bin/spark-submit
--jars cupid/hadoop-aliyun-package-3.0.0-alpha2-odps-jar-with-dependencies.jar
--class com.aliyun.odps.spark.examples.mllib.KmeansModelSaveToOss
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

1.14.6.8. PySpark interactive execution example

This topic provides an example of PySpark interactive execution in Spark on MaxCompute.

PySpark can run only on a host that can be directly connected to a computing cluster.

Install Python 2.7 and set a path:

```
export PATH=/path/to/python/bin:$PATH
```

Run the following start command:

```
bin/pyspark --master yarn-client
```

Perform interactive execution:

```
df=spark.sql("select * from spark_user_data")
df.show()
```

1.14.6.9. Spark-shell interactive execution example (read tables)

This topic provides an example of Spark-shell interactive execution (read tables) in Spark on MaxCompute.

Spark-shell can run only on a host that can be directly connected to a computing cluster.

Run the following start command:

```
bin/spark-shell --master yarn
```

Perform interactive execution:

```
sc.parallelize(0 to 100, 2).collect
sql("show tables").show
sql("select * from spark_user_data").show(200,100)
```

1.14.6.10. Spark-shell interactive execution example (Spark MLLib and OSS read/write)

This topic provides an example of Spark-shell interactive execution (Spark MLLib and OSS read/write) in Spark on MaxCompute.

Spark-shell can run only on a host that can be directly connected to a computing cluster.

Add the following configuration to conf/spark-defaults.conf:

```
spark.hadoop.fs.oss.accessKeyId=***
spark.hadoop.fs.oss.accessKeySecret=***
spark.hadoop.fs.oss.endpoint=***
```

Run the following start command:

```
bin/spark-shell --master yarn --jars cupid/hadoop-aliyun-package-3.0.0-alpha2-odps-jar-with-dependencies.jar
```

Perform interactive execution:

```
import org.apache.spark.mllib.clustering.KMeans._
import org.apache.spark.mllib.clustering.{ KMeans, KMeansModel}
import org.apache.spark.mllib.linalg.Vectors
val modelOssDir = "oss://your_bucket/kmeans-model"
val points = Seq(
  Vectors.dense(0.0, 0.0),
  Vectors.dense(0.0, 0.1),
  Vectors.dense(0.1, 0.0),
  Vectors.dense(9.0, 0.0),
  Vectors.dense(9.0, 0.2),
  Vectors.dense(9.2, 0.0)
)
val rdd = sc.parallelize(points, 3)
val initMode = K_MEANS_PARALLEL
val model = KMeans.train(rdd, k = 2, maxIterations = 2, runs = 1, initMode)
val predictResult1 = rdd.map(feature => "cluster id: " + model.predict(feature) + " feature:" + feature.toArray.mkString(",")).collect
println("modelOssDir=" + modelOssDir)
model.save(sc, modelOssDir)
val modelLoadOss = KMeansModel.load(sc, modelOssDir)
val predictResult2 = rdd.map(feature => "cluster id: " + modelLoadOss.predict(feature) + " feature:" + feature.toArray.mkString(",")).collect
assert(predictResult1.size == predictResult2.size)
predictResult2.foreach(result2 => assert(predictResult1.contains(result2)))
```

1.14.6.11. Example of SparkR interactive execution

This topic provides an example of the SparkR interactive execution in Spark on MaxCompute.

SparkR can be run only on a machine that can directly connect to a computing cluster. In addition, you must install R in the `/home/admin/R` directory and specify the path:

```
export PATH=/home/admin/R/bin:$PATH
```

The following command is used to start SparkR.

```
bin/sparkR --master yarn --archives ./R/R.zip
```

The following commands are run in interactive mode.

```
df <- as.DataFrame(faithful)
df
head(select(df, df$eruptions))
head(select(df, "eruptions"))
head(filter(df, df$waiting < 50))
results <- sql("FROM spark_user_data SELECT *")
head(results)
```

1.14.6.12. Example of PageRank with Apache Spark GraphX

This topic provides an example of PageRank with Apache Spark GraphX in Spark on MaxCompute.

Spark on MaxCompute supports Apache Spark GraphX.

Example:

```
package com.aliyun.odps.spark.examples.graphx
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.graphx._
import org.apache.spark.rdd.RDD
object PageRank {
  def main(args: Array[String]): Unit = {
    val conf = new SparkConf().setAppName("pagerank")
    val sc = new SparkContext(conf)
    // construct vertices
    val users: RDD[(VertexId, Array[String])] = sc.parallelize(List(
      "1,BarackObama,Barack Obama",
      "2,ladygaga,Goddess of Love",
      "3,jeresig,John Resig",
      "4,justinbieber,Justin Bieber",
      "6,matei_zaharia,Matei Zaharia",
      "7,odersky,Martin Odersky",
      "8,anonsys"
    ).map(line => line.split(",")).map(parts => (parts.head.toLong, parts.tail)))
    // construct edges
    val followers: RDD[Edge[Double]] = sc.parallelize(Array(
      Edge(2L,1L,1.0),
      Edge(4L,1L,1.0),
      Edge(1L,2L,1.0),
      Edge(6L,3L,1.0),
      Edge(7L,3L,1.0),
      Edge(7L,6L,1.0),
      Edge(6L,7L,1.0),
      Edge(3L,7L,1.0)
    ))
    // construct graph
    val followerGraph: Graph[Array[String], Double] = Graph(users, followers)
    // restrict the graph to users with usernames and names
    val subgraph = followerGraph.subgraph(vpred = (vid, attr) => attr.size == 2)
    // compute PageRank
    val pageRankGraph = subgraph.pageRank(0.001)
    // get attributes of the top pagerank users
    val userInfoWithPageRank = subgraph.outerJoinVertices(pageRankGraph.vertices) {
      case (uid, attrList, Some(pr)) => (pr, attrList.toList)
      case (uid, attrList, None) => (0.0, attrList.toList)
    }
    println(userInfoWithPageRank.vertices.top(5)(Ordering.by(_._2._1)).mkString("\n"))
  }
}
```

Run the following command to submit a job:

```
bin/spark-submit \
--master yarn-cluster \
--class com.aliyun.odps.spark.examples.graphx.PageRank \
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar
```

1.14.6.13. Example of Spark Streaming-NetworkWordCount

This topic provides an example of Spark Streaming-NetworkWordCount in Spark on MaxCompute.

Native Spark Streaming is supported. To use NetworkWordCount, you must install netcat on your on-premises machine, and then run the following command:

```
$ nc -lk 9999
```

The input in the command terminal becomes the input of Spark Streaming.

Example:

```
package com.aliyun.odps.spark.examples.streaming
import org.apache.spark.SparkConf
import org.apache.spark.examples.streaming.StreamingExamples
import org.apache.spark.storage.StorageLevel
import org.apache.spark.streaming.{Seconds, StreamingContext}
object NetworkWordCount {
  def main(args: Array[String]) {
    if (args.length < 2) {
      System.err.println("Usage: NetworkWordCount <hostname> <port>")
      System.exit(1)
    }
    StreamingExamples.setStreamingLogLevels()
    // Create the context with a 1 second batch size
    val sparkConf = new SparkConf().setAppName("NetworkWordCount")
    val ssc = new StreamingContext(sparkConf, Seconds(1))
    // Create a socket stream on target ip:port and count the
    // words in input stream of \n delimited text (eg. generated by 'nc')
    // Note that no duplication in storage level only for running locally.
    // Replication necessary in distributed scenario for fault tolerance.
    val lines = ssc.socketTextStream(args(0), args(1).toInt, StorageLevel.MEMORY_AND_DISK_SER)
    val words = lines.flatMap(_.split(" "))
    val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
    wordCounts.print()
    ssc.start()
    ssc.awaitTermination()
  }
}
```

Run the following command to submit a job:

```
bin/spark-submit \
--master local[4] \
--class com.aliyun.odps.spark.examples.streaming.NetworkWordCount \
/path/to/aliyun-cupid-sdk/examples/spark-examples/target/spark-examples_2.11-1.0.0-SNAPSHOT-shaded.jar localhost 9999
```

1.14.7. Maven dependencies

This topic describes Maven dependencies of Spark on MaxCompute.

The GitHub project described in the Common cases topic can be used as a template for your quick start. For custom development, use the following pom.xml file.

 **Note** Make sure that the Spark community edition is 2.3.0 and the scope is provided.

```
<dependency>
  <groupId>org.apache.spark</groupId>
  <artifactId>spark-core_2.11</artifactId>
  <version>2.3.0</version>
  <scope>provided</scope>
</dependency>
```

The MaxCompute plug-in has been released to the Maven repository. You must add the following dependencies to the pom.xml file:

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-core_2.11</artifactId>
  <version>1.0.0</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-client_2.11</artifactId>
  <version>1.0.0</version>
</dependency>
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-datasource_2.11</artifactId>
  <version>1.0.0</version>
</dependency>
```

Dependencies in the Maven repository:

- Core code of the Cupid platform: encapsulates the APIs that are used to submit Cupid tasks and the APIs that parent and child processes use to read data from and write data to tables.

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-core_2.11</artifactId>
  <version>1.0.0</version>
</dependency>
```

- Datasource: encapsulates Spark-related APIs that are used to read data from and write data to MaxCompute tables.

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-datasource_2.11</artifactId>
  <version>1.0.0</version>
```

- Client: encapsulates an SDK in Cupid client mode.

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>cupid-client_2.11</artifactId>
  <version>1.0.0</version>
</dependency>
```

1.14.8. Special notes

1.14.8.1. Running mode

This topic describes the running mode of Spark on MaxCompute.

Spark on MaxCompute supports three running modes: local, cluster, and DataWorks.

Local mode

The local mode facilitates code debugging for applications. In local mode, you can use Spark on MaxCompute the same way as that described in the Apache Spark community. You can also use Tunnel to read and write data from and to MaxCompute tables. In this mode, you can use either an integrated development environment (IDE) or the command line to run Spark on MaxCompute.

If Spark on MaxCompute works in this mode, you must add the configuration `spark.master=local[N]`. `N` indicates the CPU cores required to implement this mode.

In local mode, table read and write operations are implemented by using Tunnel. Therefore, you must add a Tunnel configuration item to the `spark-defaults.conf` file.

 **Note** Enter the endpoint based on the region and network environment in which your MaxCompute project resides.

The following code provides an example on how to use the command line to run Spark on MaxCompute in local mode:

```
1.bin/spark-submit --master local[4] \
--class com.aliyun.odps.spark.examples.SparkPi \
${path to aliyun-cupid-sdk}/spark/spark-2.x/spark-examples/target/spark-examples_2.11-version-shaded.jar
```

Cluster mode

In cluster mode, you must specify the `Main` method as the entry point of a custom application. A Spark job ends when `Main` succeeds or fails. This mode is suitable for offline jobs. You can use Spark on MaxCompute in this mode with DataWorks to schedule jobs.

The following code provides an example on how to use the command line to run Spark on MaxCompute in cluster mode:

```
1.bin/spark-submit --master yarn-cluster \
--class SparkPi \
${ProjectRoot}/spark/spark-2.x/spark-examples/target/spark-examples_2.11-version-shaded.jar
```

DataWorks mode

You can run offline jobs of Spark on MaxCompute that is in cluster mode in DataWorks to integrate and schedule other types of nodes.

Procedure:

1. Upload the resources in the DataWorks business flow and click **Submit**.
2. In the created business flow, select **ODPS Spark** from **Data Analytics**.
3. Double-click the Spark node and define the Spark job.

Select a Spark version, a development language, and a resource file for the job. The resource file is the JAR file that is uploaded and published in the business flow.

You can specify configuration items, such as the number of executors and the memory size, for the job that you want to submit.

You must also specify the endpoint configuration item `spark.hadoop.odps.cupid.webproxy.endpoint` of Spark on MaxCompute.

 **Note** Enter the endpoint of the region in which your MaxCompute project resides, for example, `http://service.cn.maxcompute.aliyun-inc.com/api`.

4. You can manually run the Spark node to view the task log and obtain the URLs of Logview and Jobview from the log for further analysis and diagnosis.
5. After the Spark job is defined, orchestrate and schedule services of different types in the business flow as required.

1.14.8.2. Spark Streaming tasks

This topic describes the configurations that are required to run Spark Streaming tasks.

MaxCompute supports Spark Streaming. To support Spark Streaming tasks that run for a long period of time, you must add the following special configurations to the `spark-defaults.conf` file:

```
spark.hadoop.odps.cupid.engine.running.type=longtime
# Set the running type of a task to longtime so that the task is not reclaimed.
spark.hadoop.odps.cupid.job.capability.duration.hours=25920
# Specify the running duration.
spark.yarn.maxAppAttempts=10
# Specify the maximum number of retries for a failover.
spark.streaming.receiver.writeAheadLog.enable=true
# Determine whether to enable the write-ahead logging mode. This feature prevents data loss but reduces data processing efficiency.
```

1.14.8.3. Job diagnosis

This topic describes job diagnosis of Spark on MaxCompute.

After you submit a job, you must check the job log to determine whether the job is submitted and executed as expected. MaxCompute provides Logview and Spark web UI for you to diagnose jobs.

The following example demonstrates how to submit a job in `spark-submit` mode. Logs are also generated when you use DataWorks to run Spark jobs.

```
cd $SPARK_HOME
bin/spark-submit --master yarn-cluster --class SparkPi /tmp/spark-2.x-demo/target/Alispark-2.x-quickstart-1.0-SNAPSHOT-shaded.jar
```

After the job is submitted, MaxCompute creates an instance and displays the Logview URL of the instance in the log.

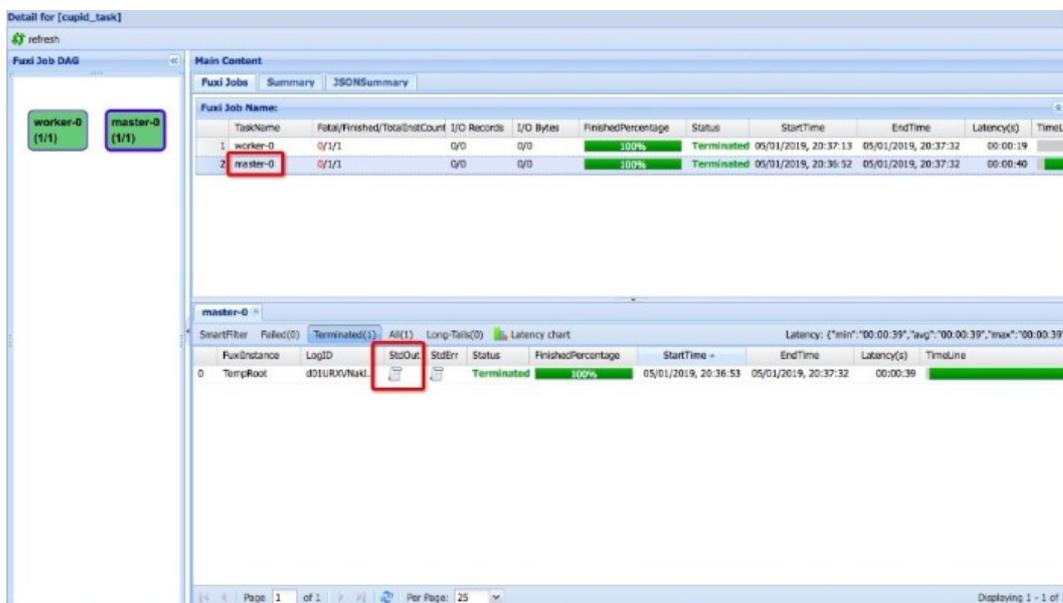
```
19/01/05 20:36:47 INFO YarnClientImplUtil: logview url: http://logview.odps.aliyun.com/logview/?h=http://service.cn.maxcompute.aliyun.com/api&p=qn_beijing&i=xxx&token=xxx
-- <The operation succeeds if an output similar to the following result is displayed.>
19/01/05 20:37:34 INFO Client:
client token: N/A
diagnostics: N/A
ApplicationMaster host: 11.220.xxx.xxx
ApplicationMaster RPC port: 30002
queue: queue
start time: 1546691807945
final status: SUCCEEDED
tracking URL: http://jobview.odps.aliyun.com/proxyview/jobview/?h=http://service.cn.maxcompute.aliyun-inc.com/api&p=project_name&i=xxx&t=spark&id=application_xxx&metaname=xxx&token=xxx
```

Use Logview to diagnose a job

1. View basic execution information about the task of the CUPID type in a browser based on the Logview URL.



2. Click the progress bar of the task whose TaskName is master-0. In the lower pane, click All and find TempRoot in the FuxiInstance column.



3. Click the icon in the StdOut column that corresponds to TempRoot to view the output of SparkPi.



Use Spark web UI to diagnose a job

The tracking URL in the log indicates that your job is submitted to the MaxCompute cluster. This URL is crucial because it is the URL of both Spark web UI and History Server.

1. Access the URL in a browser to track the running status of your Spark job.

The screenshot shows the Spark web UI interface. At the top, there are navigation tabs: Jobs, Stages, Storage, Environment, and Executors. The 'Executors' tab is selected. Below the navigation, there is a 'Summary' section with a table showing overall statistics for Active, Dead, and Total executors. Below that is a table of individual executors. The first row is for the 'driver' executor, which is highlighted with a red box. The 'Logs' column for the driver is also highlighted with a red box, showing 'stderr' and 'stdout' links. The second row is for a worker executor.

Active	Dead	Total
2	0	2

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Logs
driver	cupid-11-220-203-36:45885	Active	0	0.0 B / 2.1 GB	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B	stderr stdout
1	worker50dcb649-ce08-4151-a605-0b2034858fa3cupid-11-220-216-77:43705	Active	0	0.0 B / 3.2 GB	0.0 B	2	0	0	2	2	2 s (0.1 s)	0.0 B	0.0 B	0.0 B	stderr stdout

2. Click StdOut in the Logs column that corresponds to the driver to view the output of the Spark job.

The screenshot shows a browser window with the URL 'jobview.odps.aliyun.com/logsview/'. The page content displays 'Pi is roughly 3.1434'.

1.14.9. APIs supported by Spark

1.14.9.1. Spark Shell

This topic describes the Spark shell API of Spark on MaxCompute.

Run the following commands to start the application of the Spark shell API:

```
$cd $SPARK_HOME
-- Go to the Spark shell directory.
$bin/spark-shell --master yarn
-- Specify the running mode and start the application.
```

Example:

```
sc.parallelize(0 to 100, 2).collect
sql("show tables").show
sql("select * from spark_user_data").show(200,100)
```

1.14.9.2. Spark R

This topic describes the SparkR API of Spark on MaxCompute.

Run the following commands to start the application of the SparkR API:

```
$mkdir -p /home/admin/R && unzip ./R/R.zip -d /home/admin/R/
# Create the R directory and decompress the R.zip package in this directory.
$export PATH=/home/admin/R/bin/:$PATH
-- Configure environment variables.
$bin/sparkR --master yarn --archives ./R/R.zip
-- Specify the running mode and start the application.
```

Example:

```
df <- as.DataFrame(faithful)
df
head(select(df, df$eruptions))
head(select(df, "eruptions"))
head(filter(df, df$waiting < 50))
results <- sql("FROM spark_user_data SELECT *")
head(results)
```

1.14.9.3. Spark SQL

This topic describes the Spark SQL API of Spark on MaxCompute.

Run the following commands to start the application of the Spark SQL API:

```
$cd $SPARK_HOME
-- Go to the Spark SQL directory.
$bin/spark-sql --master yarn
-- Specify the running mode and start the application.
```

Example:

```
show tables;
select * from spark_user_data limit 3;;
quit;
```

1.14.9.4. Spark JDBC

This topic describes the Spark Java Database Connectivity (JDBC) API of Spark on MaxCompute.

Run the following commands to start the application of the Spark JDBC API:

```
$sbin/stop-thriftserver.sh
-- Stop a thread.
/sbin/start-thriftserver.sh
-- Restart a thread.
$bin/beeline
-- Start the application.
```

Example:

```
!connect jdbc:hive2://localhost:10000/odps_smoke_test
show tables;
select * from mr_input limit 3;
!quit
```

1.14.10. Dynamic resource allocation of Spark

This topic describes the dynamic resource allocation (DRA) feature of Spark on MaxCompute.

Background information

Spark provides a large number of parameters to implement a wide range of semantics. You can use the default values for most parameters, but some parameters require manual configurations. The `spark.executor.instances` parameter is the most complex.

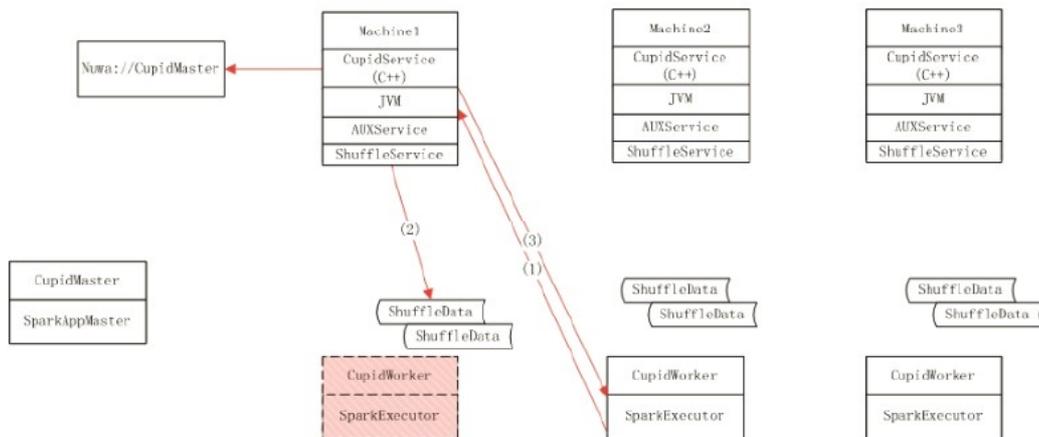
If this parameter is set to a small value, operations may run slowly or fail due to out of memory (OOM) or no free disk space. If this parameter is set to a large value, resources are wasted.

Even the optimal value obtained based on full understanding of the data and logic is not reliable. For complex jobs, the number of executors required at different stages is different and different resources are required during the job execution process. Fixed configurations of resources cause waste. If long tail latency occurs, idle resources are occupied by other executors even for simple jobs.

Solution

The best solution is to allocate resources as required. DRA is a solution that addresses this issue. CupidService developed by the Cupid team supports native DRA. The following figure shows the implementation process of DRA.

Note For more information about the native DRA in the Apache Spark community, see [Spark help](#) and [Spark configuration](#).



Enable DRA

You must add the following configurations to enable DRA. You do not need to modify the code.

```
spark.hadoop.odps.cupid.shuffle.service.enable=true // Required. This parameter specifies whether to enable the shuffle service in CupidService.
spark.hadoop.odps.cupid.disk.driver.enable=true // Required. This parameter is a dependent item.
spark.dynamicAllocation.enabled=true // Required. This parameter specifies whether to enable DRA in Spark.
spark.shuffle.service.enabled=true // Required. This parameter is a dependent item.
spark.shuffle.service.port=7338 // Required. This parameter indicates the port used for the shuffle service.
spark.authenticate=true // Required. This parameter specifies whether to enable authentication.
spark.dynamicAllocation.maxExecutors=128 // Optional. This parameter indicates the maximum number of executors.
spark.dynamicAllocation.minExecutors=1 // Optional. This parameter indicates the minimum number of executors.
spark.dynamicAllocation.initialExecutors=1 // Optional. This parameter indicates the initial number of executors.
spark.dynamicAllocation.executorIdleTimeout=60s // Optional. This parameter indicates the waiting period before idle executors are released.
```

If DRA is enabled, `spark.executor.instances` is optional and equivalent to `spark.dynamicAllocation.initialExecutors`.

1.14.11. FAQ of Spark on MaxCompute

This topic provides answers to some frequently asked questions about Spark on MaxCompute.

How do I migrate code of the open source Spark to Spark on MaxCompute?

The migration method depends on whether your job needs to access MaxCompute tables or Object Storage Service (OSS):

- If the job does not need to access MaxCompute tables or OSS, run your JAR package directly. Note that you must set the dependency of Spark or Hadoop to provided.
- If the job needs to access MaxCompute tables, you need only to configure related dependencies and repackage them.
- If the job needs to access OSS, you need only to configure related dependencies and repackage them.

How do I use Spark on MaxCompute to access services in a VPC?

Spark on MaxCompute does not allow access to services in a VPC. If you want to access a service in a VPC, submit a ticket to contact the MaxCompute technical support team.

The following error message appears, which indicates that the ID and key in the `spark-defaults.conf` file are invalid. What do I do?

Error:

```
Stack:
com.aliyun.odps.OdpsException: ODPS-0410042:
Invalid signature value - User signature dose not match
```

Check whether the ID and key in the `spark-defaults.conf` file are consistent with the AccessKey ID and AccessKey secret that you obtain from the Apsara Uni-manager Management Console.

The following error message appears, which indicates that I do not have permissions. What do I do?

Error:

```
Stack:
com.aliyun.odps.OdpsException: ODPS-0420095:
Access Denied - Authorization Failed [4019], You have NO privilege 'odps:CreateResource' on {acs:odps*:projects/*}
```

The project owner must grant the READ and CREATE permissions on resources to you.

The following error message appears, which indicates that the project does not support Spark jobs. What do I do?

Error:

```
Exception in thread "main" org.apache.hadoop.yarn.exceptions.YarnException: com.aliyun.odps.OdpsException: ODPS-0420095: Access Denied - The task is not in release range: CUPID
```

Check whether the Spark on MaxCompute service is provided in the region where the project resides. In addition, check whether the configurations in the `spark-defaults.conf` file are consistent with those described in the MaxCompute documentation. If the Spark on MaxCompute service is provided in the region and the configurations in the `spark-defaults.conf` file are correct, submit a ticket.

When a task is running, the following error message appears, which indicates that local storage space is insufficient. What do I do?

Error:

```
No space left on device
```

Spark on MaxCompute uses online storage to replace local storage. Shuffled data and overflow data of BlockManager are all stored online. Therefore, you must check the setting of the online storage space. The storage space is determined by the `spark.hadoop.odps.cupid.disk.driver.device_size` parameter. The default storage space is 20 GB and the maximum storage space is 100 GB. If the error persists after you increase the storage space to 100 GB, conduct further analysis.

The most common cause is data skew. Data is unevenly distributed among blocks during the shuffle and cache processes. If this is the case, you can decrease the number of concurrent tasks in each executor (`spark.executor.cores`) or increase the number of executors (`spark.executor.instances`).

1.15. Elasticsearch on Maxcompute

1.15.1. Overview

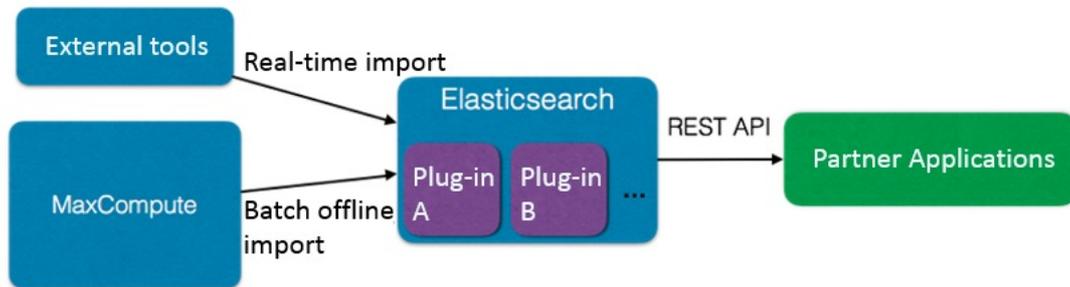
This topic provides an overview of Elasticsearch on MaxCompute and its core features.

Elasticsearch on MaxCompute is an enterprise-class full-text index system developed by Alibaba Cloud to retrieve large amounts of data. It provides near-real-time (NRT) search for government agencies and enterprises. It has the following benefits:

- Provides scalable full-text index services and supports native Elasticsearch API operations.
- Supports data import from multiple heterogeneous data sources based on the API development.
- Supports cluster- and business-level O&M management.

As a combination of Elasticsearch and MaxCompute, Elasticsearch on MaxCompute provides efficient, core massive data search engine services by leveraging unified scheduling and management of MaxCompute. In addition, Elasticsearch on MaxCompute can be used with the plug-ins of open source Elasticsearch to provide a wide range of index features.

Elasticsearch on MaxCompute allows you to use tools to import data from external sources in real time. You can also import offline data from MaxCompute. After the imported data is indexed, Elasticsearch on MaxCompute provides index services by using RESTful APIs. The following figure shows the usage of Elasticsearch on MaxCompute.



1.15.2. Workflow

1.15.2.1. Overview

This topic describes the workflow of Elasticsearch on MaxCompute.

Elasticsearch on MaxCompute is developed based on the open source Elasticsearch. It can run the Elasticsearch service on MaxCompute clusters.

In the MaxCompute client, you can start and manage your Elasticsearch service as required, including the number of nodes, disk space, memory size, and custom settings. The resources consumed by Elasticsearch are counted towards your MaxCompute quota.

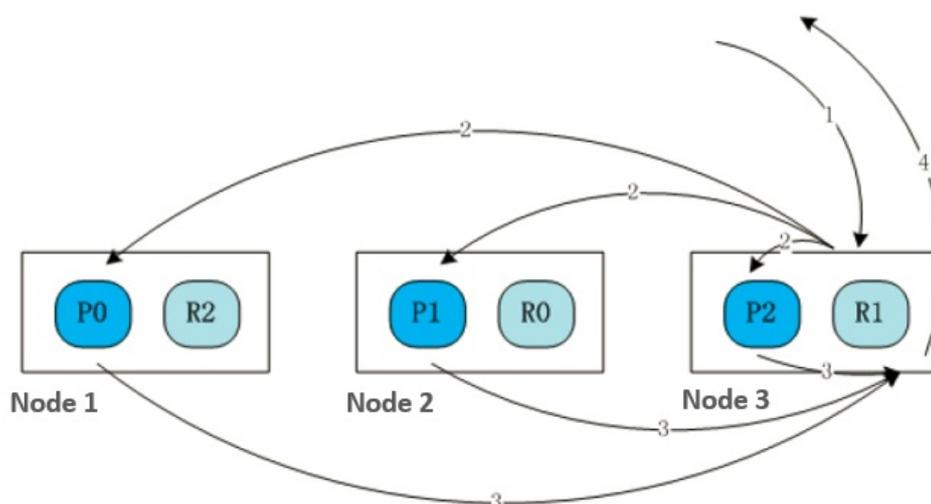
For more information about the process for starting Elasticsearch, see [Elasticsearch typical practice](#).

The following topics describe the workflows of the features of Elasticsearch on MaxCompute.

1.15.2.2. Distributed search workflow

This topic describes the distributed search workflow of Elasticsearch on MaxCompute.

The following figure shows the distributed search workflow.



In the preceding figure, the cluster consists of three nodes. The index has three shards: P0, P1, and P2. These shards are distributed across the three nodes. Each shard is replicated in 1:1 mode. Three replicas are generated: R0, R1, and R2.

1. A user sends a search request to Node 3.
2. After Node 3 receives the request, it sends a search request (2) to P0, P1, and P2 based on the recorded index shard information.
3. The nodes in which P0, P1, and P2 are located search for the requested information in the specified shards. A search result message (3) is sent to Node 3.
4. Node 3 collects the search results from other nodes and returns the search results to the user in an acknowledgment message (4).

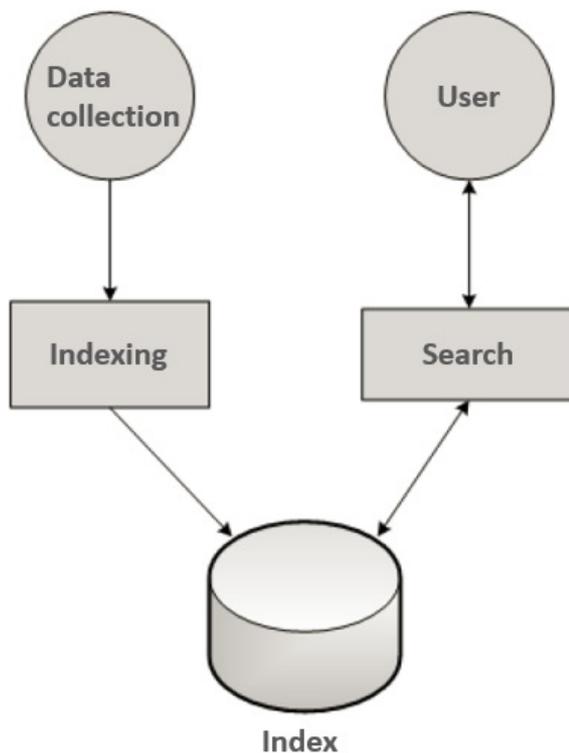
Note

- The search speed is increased because multiple nodes perform the search at the same time.
- The performance of distributed search improves with the increase of nodes.

1.15.2.3. Full-text index workflow

This topic describes the full-text index workflow of Elasticsearch on MaxCompute.

The following figure shows the full-text index workflow.



1. The data collection module collects both structured and unstructured data, converts the data into the field+value format, and submits the data to the indexing module.
2. After the indexing module receives the field+value data, it tokenizes and creates an inverted index based on the predefined indexing method of the field and saves the data to the index. The type, indexing method, and tokenization method of each field are configured on the retrieval management platform.

3. A user sends a search request. After the search processing module receives and processes the request, the search index, field, and query statement are obtained. The specified record list is included in the inverted index data.
4. The indexing module returns data that meets the requirements such as sorting rules and the number of requests.

1.15.2.4. Authentication process

This topic describes the authentication process of Elasticsearch on MaxCompute.

Authentication process:

1. Elasticsearch on MaxCompute allows you to perform retrieval management and O&M on MaxCompute. To perform these operations, you must log on to the retrieval management or O&M platform that is provided by Elasticsearch on MaxCompute. During logon, you are redirected to the authentication module for authentication. If the authentication fails, you are not allowed to access the related platform.
2. The administrator can use the MaxCompute client to add Elasticsearch users and grant permissions for the users.
3. The system authenticates all users who attempt to access index libraries. After you pass the authentication, you are allowed to retrieve data or perform operations on data in the libraries.

1.15.3. Quick start

This topic describes how to use Elasticsearch on MaxCompute. It guides you through the basic use of Elasticsearch on MaxCompute.

Before you start an Elasticsearch cluster, make sure that you have determined the following information:

- **Node planning:** Determine the number of nodes that are required for each role in an Elasticsearch cluster. By default, an Elasticsearch cluster has two roles, master and data. Each role is deployed on three nodes. You can add nodes to the cluster at any time.
- **Resource planning:** Determine the vCPU, memory, and disk space resources that are required for each node. The resources configured for each node cannot be changed. By default, each node is assigned 8 GB of memory and 20 GB of disk space.

 **Note** Only 50% of the memory allocated to a data node is used for the JVM heap.

- **Elasticsearch configuration:** Determine the running configurations of nodes in an Elasticsearch cluster, such as the queue size of bulk requests, and support for cross-domain HTTP requests.

After you determine the preceding information, you can start your Elasticsearch cluster in the MaxCompute client.

The following example demonstrates how to quickly start a small Elasticsearch cluster based on the default configuration. In the following example, the name of the Elasticsearch cluster you want to start is `es_first_cluster`.

1. Download the [MaxCompute client](#) that supports Elasticsearch. Configure the AccessKey pair, project, and endpoint.
2. Run `odpscmd` and run the following command to start the Elasticsearch cluster:

```
server create es_first_cluster type elasticsearch_mdu;
```

Wait for several minutes. If OK is returned, the Elasticsearch cluster is started. You must create an Elasticsearch user to access the cluster.

3. In `odpscmd`, create an Elasticsearch user with the `ALL_ACCESS` permission.

```
server execute es_first_cluster create user admin with password 123456|all_access;
```

If OK is returned, the Elasticsearch user is created.

4. Access the started Elasticsearch cluster. In the following example, the MaxCompute project that is used to access the Elasticsearch cluster is prj1. Run the following command to return information about the Elasticsearch cluster:

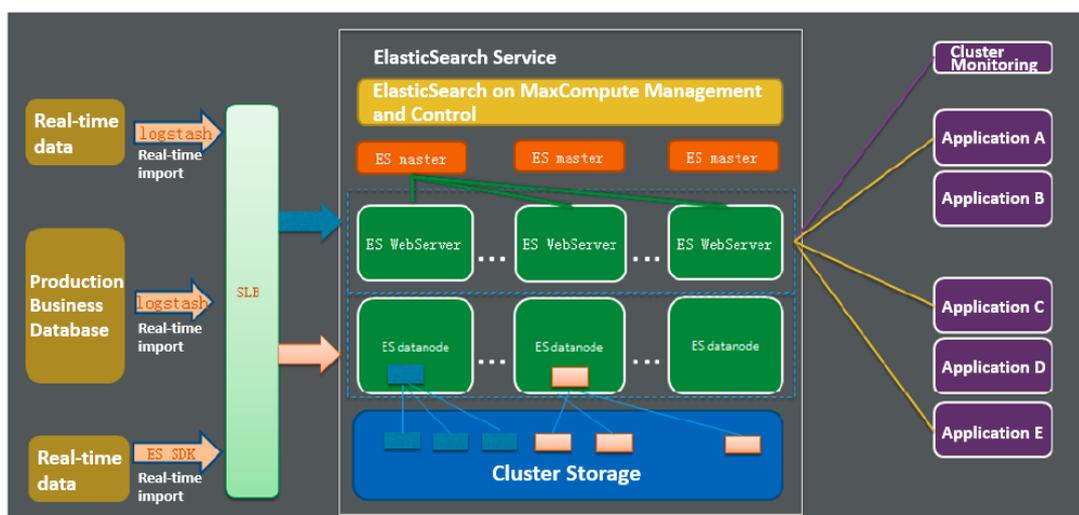
```
curl -u admin:123456 http://search.aliyun.com:9200/prj1.es_first_cluster;
```

Notice To delete the Elasticsearch cluster, run the `server delete es_first_cluster` command. If you run this command, the Elasticsearch cluster is permanently deleted, and data cannot be restored. We recommend that you proceed with caution.

1.15.4. Support for Elasticsearch applications

1.15.4.1. Typical practice of Elasticsearch on MaxCompute

This topic describes the typical practice of Elasticsearch on MaxCompute.



Elasticsearch on MaxCompute allows you to start an Elasticsearch cluster in a MaxCompute cluster by submitting a job. MaxCompute projects do not modify native Elasticsearch code. The Elasticsearch on MaxCompute runs in the same mode as native Elasticsearch clusters.

1.15.4.2. Limits on Elasticsearch on MaxCompute in VPCs

This topic describes the limits on Elasticsearch on MaxCompute in VPCs.

Elasticsearch on MaxCompute is an enterprise-class retrieval system that is developed by Alibaba Cloud to retrieve large amounts of data. This system also complies with data isolation and security requirements. Therefore, the limits on Elasticsearch on MaxCompute in VPCs are imposed based on the limits of MaxCompute in VPCs.

Elasticsearch on MaxCompute has the following limits in VPCs:

- The classic network, VPCs, and the Internet are isolated from each other. Users can only access the endpoints and virtual IP addresses (VIPs) of their own networks.
- Projects for which VPC IDs or IP address whitelists are not configured are accessible to the three types of

networks by using domain names.

- If an Elasticsearch cluster is started in a MaxCompute project, the cluster and project share the same VPC list. The VPC list is a whitelist of VPCs.
- If you start an Elasticsearch cluster, this cluster automatically occupies all resources. If you start more Elasticsearch instances, you must scale up the MaxCompute instance or scale down the Elasticsearch cluster.

Scenario: When you deploy MaxCompute in Apsara Stack, one project is created and one Elasticsearch cluster is started for each project by default. You can start your own Elasticsearch cluster in your project, apply for a domain name and VIP after the cluster is started, and then perform VPC verification in the Elasticsearch frontend.

1.15.5. Special notes

1.15.5.1. Find the Elasticsearch service domain name

This topic describes how to find the domain name of the Elasticsearch service during O&M.

1. Find your MaxCompute cluster in the Apsara Infrastructure Management Framework.
2. Navigate to the MaxCompute cluster resource usage page.
3. Find the domain name of the Elasticsearch service.

1.15.5.2. Import table data from MaxCompute to

Elasticsearch

This topic describes how to import table data from MaxCompute to Elasticsearch.

Before you use Elasticsearch on MaxCompute to search for the data in a MaxCompute table, you must import the table data from MaxCompute to an Elasticsearch cluster. To meet these requirements, Alibaba Cloud develops jobs to export data from MaxCompute to Elasticsearch based on MaxCompute MapReduce. This allows you to import data from MaxCompute to Elasticsearch by using simple configurations.

You can easily control parallelism based on the distributed scheduling capability of MaxCompute. You can also add MapReduce jobs to the scheduled tasks on DataWorks.

The following example shows the data import process:

1. Download the JAR package of a MapReduce job.
2. Run the following command to add the JAR package to the MaxCompute resource files in the MaxCompute console:

```
add jar /PATH/TO/elasticsearch_output-1.0.0.jar
```

3. Create the configuration file es_mr.conf in the following format for the MapReduce job:

```
<configuration>
  <property>
    <name>key1</name>
    <value>value1</value>
  </property>
  <property>
    <name>key2</name>
    <value>value2</value>
  </property>
</configuration>
```

4. Submit the MapReduce job in the MaxCompute console. Example:

```
jar -conf es_mr.conf -classpath /PATH/TO/elasticsearch_output-1.0.0.jar
  -resources elasticsearch_output-1.0.0.jar -Dworker_num=5
  com.aliyun.odps.export.elasticsearch.mr.EsOutputJob <TABLE_NAME> [PARTITION_SPEC];
  -- Five worker nodes run in parallel to export data from the table specified by <TABLE_NAME> [PARTITION_SPEC] to the Elasticsearch cluster.
```

The following table describes the parameters in the es_mr.conf file:

Parameter	Example value	Required	Default value	Description
es.resource	my_index/my_type	Yes	N/A	The index and type of the Elasticsearch cluster into which data is imported.
es.nodes	N/A	Yes	N/A	The endpoint of Elasticsearch.
es.nodes.client.only	True	No	False	Data is sent only to client-only nodes.
es.col.field.mapping	odps_col1:es_field1,odps_col3:es_field2	Yes	N/A	The mapping between the MaxCompute columns you want to import and the Elasticsearch fields.
es.batch.size.bytes	1 MB	No	1 MB	The size of data that is transmitted at a time.
es.batch.size.entries	1000	No	1000	The number of data entries that are transmitted at a time.
es.net.http.auth.user	Admin	No	N/A	The username that is used to access the Elasticsearch cluster.
es.net.http.auth.password	123456	No		The password that is used to access the Elasticsearch cluster.
es.mapping.routing	field_routing	No	N/A	The routing field name, which is in the <CONSTANT> format for a constant.
es.mapping.id	field_id	No	N/A	The ID field of a document.

1.16. Flink on MaxCompute

This topic provides a demo of Flink on MaxCompute.

In the current MaxCompute version, Flink on MaxCompute is only for trial use. This demo describes only the trial features. For more information about new features, see later versions.

Demo:

1. Modify the configuration file.

Decompress the Flink package, go to the configuration file directory *conf*, and make the following modifications to the *flink-conf.yaml* file:

```
project_name: xxxx
access_id: xxxx
access_key: xxxx
end_point: xxxx
odps.cupid.distributedcache.mincopy: 1
odps.cupid.proxy.domain.name:xxxx
#odps.task.major.version:
cupid_v2
```

Note

- The first four parameters can be obtained from the *odps_config.ini* file under the */home/admin/odps/odps_tools/clt/conf/* directory.
- *odps.cupid.proxy.domain.name* specifies the domain name of *odps_jobview_server_dns*. The value of this parameter is the domain name without *sparkui*.

2. Start the program.

Save the *flink-java-project-0.1.jar* package to the decompressed directory of Flink and run the following commands:

```
/bin/flink
run -c org.apache.flink.odps.WordCount -m yarn-cluster -yn 1
flink-java-project-0.1.jar --projectName odps_smoke_test --inputTable wc_in
--outputTable wc_out --sleepTime 100
```

3. Log on to Logview and Flink.

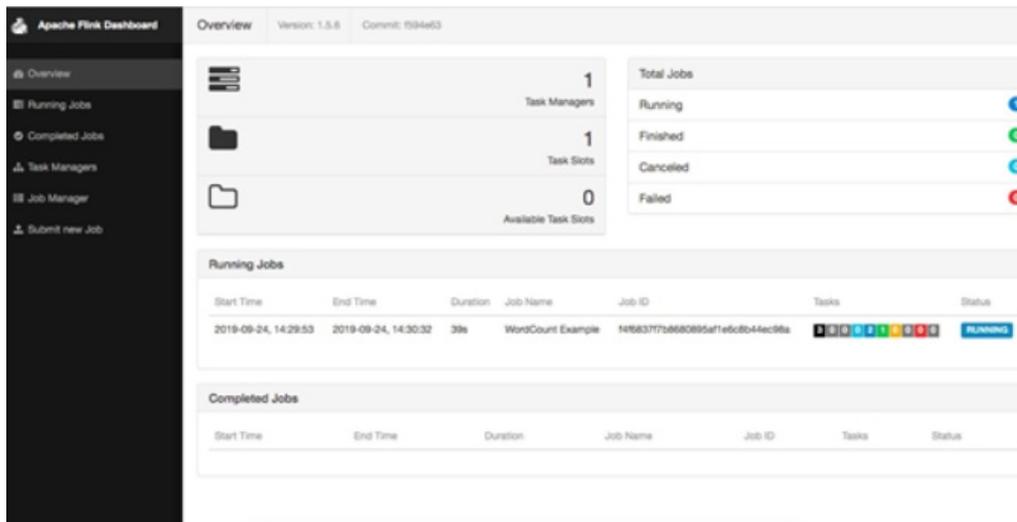
Logview

The screenshot shows the MaxCompute Logview interface. At the top, there is a table for 'ODPs Instance' with columns: URL, Project, InstanceID, Owner, StartTime, EndTime, Latency, Status, Priority, SourceID, and Tool. Below this is a 'CUPID' task icon with a 'Deploy' button. At the bottom, there is a table for 'ODPs Tasks' with columns: Name, Type, Status, Result, Detail, History, StartTime, EndTime, Latency, and TimeLine.

URL	Project	InstanceID	Owner	StartTime	EndTime	Latency	Status	Priority	SourceID	Tool
http://service...	odps_smo...	2018024062...	ALYUN@west300...	24/09/2018, 14:29:47	-	00:01:15	Running	1	flink	No Link (not internal project)

Name	Type	Status	Result	Detail	History	StartTime	EndTime	Latency	TimeLine
cupid_task	CUPID	Running				24/09/2018, 14:29:47	-	00:01:15	

Flink



Note MaxCompute is compatible with Flink in the following aspects:

- Group windows of Flink: The TUMBLE, HOP, and SESSION built-in functions are supported.
- User-defined functions (UDFs), user-defined aggregate functions (UDAFs), and user-defined table-valued functions (UDTFs) in Flink: MaxCompute supports UDFs that run in Flink. You can specify `odps.sql.enable.flink.udf` to enable this feature.
- SQL syntax in Flink: The `||` operator can be used to connect strings. `Extract(Dateunit from datetime)` and `Substring(Str from start Index to endIndex)` are supported.
- Flink HistoryServer: You can use Flink HistoryServer to view information about completed tasks.

1.17. Non-structured data access and processing (integrated computing scenarios)

1.17.1. Overview

This topic describes how to use external tables to access and process unstructured data from internal and external data sources.

MaxCompute SQL cannot directly process external data, such as non-structured data from Object Storage Service (OSS). This type of data must be imported into MaxCompute tables by using relevant tools. This involves complex operations. The MaxCompute team introduced the non-structured data processing framework to the MaxCompute system architecture to simplify the processing of external data.

You can execute a data definition language (DDL) statement to create an external table in MaxCompute and associate the table with external data sources. This table can then act as an interface between MaxCompute and external data sources. External tables can be accessed the same way as standard MaxCompute tables. You can fully use the computing capabilities of MaxCompute SQL to process external data.

MaxCompute allows you to create external tables to process data from the following data sources:

- Internal data sources: OSS, Tablestore, AnalyticDB for MySQL, ApsaraDB RDS, Apsara File Storage for HDFS, and Taobao Distributed Data Layer (TDDL)
- External data sources: open source Hadoop Distributed File System (HDFS), ApsaraDB for MongoDB, and ApsaraDB for HBase

The subsequent topics describe various data sources.

 **Note** MaxCompute V2.0 supports multiple new data types. To use the new data types, you must add `set odps.sql.type.system.odps2=true;` before the SQL statement and commit them to enable the new data types. For ease of reading, sample code in subsequent topics uses new data types by default.

1.17.2. Internal data sources

1.17.2.1. OSS data source

1.17.2.1.1. Overview

This topic describes how to access and process Object Storage Service (OSS) data sources in MaxCompute.

As a core computing component of the Alibaba Cloud big data platform, MaxCompute provides powerful computing capabilities. It can schedule a large number of nodes for parallel computing, and effectively manage the failover and retry mechanisms in a distributed computing environment. MaxCompute SQL implements a variety of logic to process data based on simple semantics. It is widely used within and outside Alibaba Group. It allows interoperability among different data sources and is crucial for building a data ecosystem for Alibaba Cloud.

Some examples are provided to demonstrate how MaxCompute accesses and processes OSS data.

1.17.2.1.2. Use a built-in extractor to read data from OSS

1.17.2.1.2.1. Overview

This topic describes how to use a built-in extractor of MaxCompute to read data from Object Storage Service (OSS).

You can use the built-in extractor of MaxCompute. This provides an easy way to read data that is stored in the specified format from OSS. You need only to create an external table as a source table for data queries.

For example, a CSV file is stored in OSS. The endpoint is `oss-cn-shanghai-internal.aliyuncs.com`, the bucket is `oss-odps-test`, and the data file is saved in `/demo/SampleData/CSV/AmbulanceData/vehicle.csv`. Some examples are provided to demonstrate how to use the built-in extractor to read data from OSS.

1.17.2.1.2.2. Create an external table

This topic describes how to create an external table when a built-in extractor is used. It also provides an example for reference.

Execute the following statements to create an external table:

```
CREATE EXTERNAL TABLE IF NOT EXISTS ambulance_data_csv_external
(
  vehicleId INT,
  recordId INT,
  patientId INT,
  calls INT,
  locationLatitude DOUBLE,
  locationLongitude DOUBLE,
  recordTime STRING,
  direction STRING
)
STORED BY 'com.aliyun.odps.CsvStorageHandler'
WITH SERDEPROPERTIES (
  'odps.properties.rolearn'='acs:ram::xxxxx:role/aliyunodpsdefaultrole'
)
LOCATION 'oss://oss-cn-shanghai-internal.aliyuncs.com/oss-odps-test/Demo/';
```

Note

- **STORED BY:** specifies the name of the built-in storage handler. `com.aliyun.odps.CsvStorageHandler` is a built-in storage handler that processes CSV files. It defines how to read data from and write data to CSV files. You can specify this parameter as required. The logic of reading and writing CSV files is implemented by the system.
- **WITH SERDEPROPERTIES:** specifies the properties of the external table that you want to create. `odps.properties.rolearn` specifies the Alibaba Cloud Resource Name (ARN) for the `AliyunODPSDefaultRole` role in Resource Access Management (RAM). You must specify this parameter if you use Security Token Service (STS) to authorize MaxCompute to access OSS.
- **LOCATION:** specifies the OSS directory in which the data files you want to read are saved. By default, the system reads all the data files in the directory.
- An external table contains only related OSS directories. If you delete this table, the data in the directory specified by `LOCATION` is not deleted.

You can execute the following statement to view the structure of the created external table:

```
DESC EXTENDED <table_name>;
```

In the output, you can view basic table information, which is similar to the information returned for a created internal table. In addition to the basic table information, you can view the storage handler and the OSS directory.

1.17.2.1.2.3. Query an external table

This topic describes how to query an external table when a built-in extractor is used. It also provides an example for reference.

After an external table is created, you can use it the same way that you use a standard table.

In this example, the `vehicle.csv` file in the `/demo/SampleData/CSV/AmbulanceData/` directory contains the following data:

```
1,1,51,1,46.81006,-92.08174,9/14/2017 0:00,S
1,2,13,1,46.81006,-92.08174,9/14/2017 0:00,NE
1,3,48,1,46.81006,-92.08174,9/14/2017 0:00,NE
1,4,30,1,46.81006,-92.08174,9/14/2017 0:00,W
1,5,47,1,46.81006,-92.08174,9/14/2017 0:00,S
1,6,9,1,46.81006,-92.08174,9/14/2017 0:00,S
1,7,53,1,46.81006,-92.08174,9/14/2017 0:00,N
1,8,63,1,46.81006,-92.08174,9/14/2017 0:00,SW
1,9,4,1,46.81006,-92.08174,9/14/2017 0:00,NE
1,10,31,1,46.81006,-92.08174,9/14/2017 0:00,N
```

Execute the following statement to submit a job, which calls a built-in CSV extractor to read data from OSS:

```
SELECT recordId, patientId, direction;
FROM ambulance_data_csv_external;
WHERE patientId > 25;
```

 **Note**

- An external table can be managed only by using MaxCompute SQL, not MaxCompute MapReduce.
- If you want to obtain data over HTTPS at the underlying layer, add `set odps.sql.unstructured.data.oss.use.https=true;` before an SQL statement. Then, commit them for execution.

The following result is returned:

```
+-----+-----+-----+
| recordId | patientId | direction |
+-----+-----+-----+
| 1 | 51 | S |
| 3 | 48 | NE |
| 4 | 30 | W |
| 5 | 47 | S |
| 7 | 53 | N |
| 8 | 63 | SW |
| 10 | 31 | N |
+-----+-----+-----+
```

 **Note** The system provides the following built-in storage handlers: CsvStorageHandler, TsvStorageHandler, and TextStorageHandler.

1.17.2.1.2.4. MSCK REPAIR TABLE

This topic describes the MSCK REPAIR TABLE statement of MaxCompute and provides an example for reference.

The MSCK REPAIR TABLE statement of MaxCompute can be used to add partitions to external tables. The following example shows the syntax of the MSCK REPAIR TABLE statement:

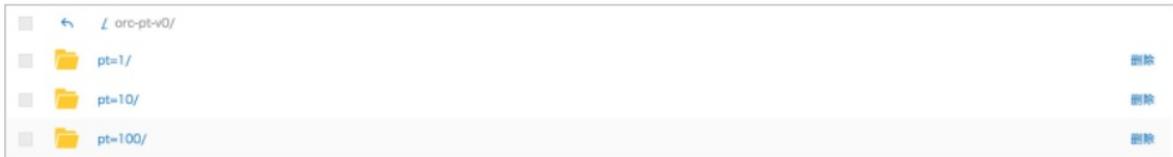
```
MSCK [REPAIR] TABLE external_table_name [ADD PARTITIONS];
```

Syntax description:

- When you import data into Object Storage Service (OSS), make sure that the OSS directory is in the `oss://xxx/table-location/ptname1=ptvalue1/ptname2=ptvalue2/xxx` format.
- You must specify the partition structure when you create an external table.
- After you execute the `MSCK [REPAIR] TABLE external_table_name [ADD PARTITIONS];` statement, MaxCompute SQL automatically parses the OSS directory to identify partitions and adds the partitions to the external table.

Example:

1. Upload data to OSS. The following figure shows the OSS directory.



2. Execute the following statement to create an external table named `orc_pt_v0`. The structure of the `pt` partition is specified in the external table.

```
CREATE EXTERNAL TABLE orc_pt_v0
(
  name STRING
)
PARTITIONED BY (pt bigint)
STORED BY 'com.aliyun.odps.CsvStorageHandler'
WITH SERDEPROPERTIES (
  'odps.properties.rolearn'='acs:ram::xxxx:role/aliyunodpsdefaultrole'
)
LOCATION 'oss://xxx/odps-ext-reg-perf/orc-pt-v0';
```

3. Execute the following statement to add three partitions to the external table `orc_pt_v0`:

```
MSCK REPAIR TABLE orc_pt_v0 ADD PARTITIONS;
-- In this case, the MSCK REPAIR TABLE statement is equivalent to the following three statements:
ALTER TABLE orc_pt_v0 ADD PARTITION (pt=1);
ALTER TABLE orc_pt_v0 ADD PARTITION (pt=10);
ALTER TABLE orc_pt_v0 ADD PARTITION (pt=100);
```

1.17.2.1.2.5. Read data from the CSV or TSV files compressed in the GZIP format

This topic describes how to read data from the CSV or TSV files that are compressed in the GZIP format.

MaxCompute can use only a built-in extractor to read data from the CSV and TSV files that are compressed in the GZIP format. The main difference between reading non-compressed and compressed files is the property specified by `SERDEPROPERTIES`.

Note If a data file stored in Object Storage Service (OSS) is an archived object, you must restore the data file first.

The following example shows how to create an external table:

```
CREATE EXTERNAL TABLE IF NOT EXISTS ambulance_data_csv_external
(
  vehicleId BIGINT,
  recordId BIGINT,
  patientId BIGINT,
  calls BIGINT,
  locationLatitude DOUBLE,
  locationLongitude DOUBLE,
  recordTime STRING,
  direction STRING
)
STORED BY 'com.aliyun.odps.CsvStorageHandler'
WITH SERDEPROPERTIES (
  'odps.properties.rolearn'='acs:ram::xxxx:role/aliyunodpsdefaultrole'
  ['odps.text.option.gzip.input.enabled'='true']
  ['name3'='value3']
)
LOCATION 'oss://oss-cn-hangzhou-zmf.aliyuncs.com/oss-odps-test/Demo/SampleData/CSV/AmbulanceData/';
```

The following table describes the properties supported by SERDEPROPERTIES.

Property	Valid value	Default value	Description
odps.text.option.gzip.input.enabled	<ul style="list-style-type: none"> • True • False 	False	Specifies whether to compress the file before data reading.
odps.text.option.gzip.output.enabled	<ul style="list-style-type: none"> • True • False 	False	Specifies whether to compress the file before data writing.
odps.text.option.header.lines.count	Non-negative integer	0	Specifies the number of rows to skip in the file.
odps.text.option.null.indicator	String	Empty string	Specifies the strings in the file to be parsed as NULL in an SQL statement. For example, if you specify \N to represent NULL, a,\N,b is parsed as a, NULL, b .
odps.text.option.ignore.empty.lines	<ul style="list-style-type: none"> • True • False 	True	Specifies whether to ignore blank rows.
odps.text.option.encoding	<ul style="list-style-type: none"> • UTF-8 • UTF-16 • US-ASCII • GBK 	UTF-8	Specifies the encoding format of the file.
odps.text.option.delimiter	Single character	Comma (,)	Specifies the column delimiter of the file.

Property	Valid value	Default value	Description
odps.text.option.use.quote	<ul style="list-style-type: none"> • True • False 	False	Specifies whether to recognize column delimiters in a CSV file if it uses double quotation marks (") as column delimiters. If fields in the CSV file contain specified symbols for separating multiple values, these fields must be enclosed in double quotation marks ("). The symbols include a carriage return, line feed (CRLF) pair, double quotation mark ("), or comma (.). If a field contains a pair of double quotation marks ("), replace the double quotation marks (") with two pairs of double quotation marks (" ") for escaping.

 **Note**

If you want to read data from and write data to an external table that is associated with compressed OSS data, configure both `odps.text.option.gzip.input.enabled` and `odps.text.option.gzip.output.enabled` as True when you create the external table.

1.17.2.1.3. Use a custom extractor to read data from OSS

1.17.2.1.3.1. Overview

This topic describes how to use a custom extractor to read data from Object Storage Service (OSS).

If OSS data is in a complex format and cannot be processed by the built-in extractor, you must use a custom extractor to read the OSS data.

For example, a text file is stored in OSS. The columns of data records in the file are separated by vertical bars (|). The file is saved in the `/demo/SampleData/CustomTxt/AmbulanceData/vehicle.csv` directory. Some examples are provided to demonstrate how to use a custom extractor to read data from OSS.

1.17.2.1.3.2. Define a storage handler

This topic describes how to define a storage handler when a custom extractor is used. It also provides an example for reference.

You can customize the logic to parse data. `StorageHandler` is the unified entrance of your custom logic. You can use `StorageHandler` to specify the types of custom extractors and `Outputer`. `StorageHandler` provides only a simple definition. For example, you can customize a `SpeicalTextStorageHandler`:

```
package com.aliyun.odps.udf.example.text;
public class SpeicalTextStorageHandler extends OdpsStorageHandler {
    @Override
    public Class<? extends Extractor> getExtractorClass() {
        return TextExtractor.class;
    }
    @Override
    public Class<? extends Outputer> getOutputerClass() {
        return TextOutputer.class;
    }
}
```

 **Note** TextStorageHandler that is built-in to MaxCompute can process the text data in which columns are separated by vertical bars (|). This example demonstrates how to customize a storage handler by using an SDK to process specially formatted data, especially in scenarios in which you use an extractor.

1.17.2.1.3.3. Define an extractor

This topic describes how to define a custom extractor when you want to use this extractor. It also provides an example for reference.

In the following example, TextExtractor is used to extract records from a text file, where the delimiter is imported as a parameter. TextExtractor can be used for all text files of the similar format.

```
/**
 * Text extractor that extract schematized records from formatted plain-
 text(csv, tsv etc.)
 **/
public class TextExtractor extends Extractor {
private InputStreamSet inputs;
private String columnDelimiter;
private DataAttributes attributes;
private BufferedReader currentReader;
private boolean firstRead = true;
public TextExtractor() {
// default to ",", this can be overwritten if a specific delimiter is
provided (via DataAttributes)
this.columnDelimiter = ",";
}
// no particular usage for execution context in this example
@Override
public void setup(ExecutionContext ctx, InputStreamSet inputs,
DataAttributes attributes) {
this.inputs = inputs;
-- inputs specifies an InputStreamSet. An InputStream is returned each time the next() method is called. This Input
Stream can read all data from an OSS file.
this.attributes = attributes;
// check if "delimiter" attribute is supplied via SQL query
String columnDelimiter = this.attributes.getValueByKey("delimiter");
-- The delimiter can be used as a parameter in DDL statements.
if ( columnDelimiter != null)
{
this.columnDelimiter = columnDelimiter;
}
// note: more properties can be inited from attributes if needed
}
@Override
public Record extract() throws IOException {
String line = readNextLine();
if (line == null) {
return null;
-- If null is returned, all records in the table have been read.
}
return textLineToRecord(line);
-- textLineToRecord splits a row into multiple columns by using the delimiter. For more information about the impl
ementation process, see Complete implementation of TextExtractor.
-- extractor() returns a record that is extracted from OSS data.
}
@Override
public void close(){
// no-op
}
}
```

1.17.2.1.3.4. Compile and package code

This topic describes how to compile and package code when you use a custom extractor.

You can compile and package Java code, and run the following command to upload the package to MaxCompute. The procedure is the same as that for a common Java user-defined function (UDF).

```
add jar odps-udf-example.jar;
```

1.17.2.1.3.5. Create an external table

This topic describes how to create an external table when a custom extractor is used. It also provides an example for reference.

After you upload a JAR package, you must run the following command to create an external table. This command is similar to that you run when a built-in extractor is used. The difference is that a custom storage handler is used in this command.

```
CREATE EXTERNAL TABLE IF NOT EXISTS ambulance_data_txt_external
(
  vehicleId int,
  recordId int,
  patientId int,
  calls int,
  locationLatitude double,
  locationLongitude double,
  recordTime string,
  direction string
)
STORED BY 'com.aliyun.odps.udf.example.text.SpecialTextStorageHandler'
-- STORED BY specifies the class name of a custom storage handler.
WITH SERDEPROPERTIES('delimiter'='|')
-- SERDEPROPERTIES can be used to specify parameters. These parameters are transferred to an extractor by using
DataAttributes.
LOCATION 'oss://<*>your-id*>:<*>your-secret-key*>@oss-cn-shanghai-internal.aliyuncs.com/oss-odps-test/Demo/Sa
mpleData/CustomTxt/AmbulanceData/'
USING 'odps-udf-example.jar';
-- Specify the JAR package in which the class definition is located.
```

1.17.2.1.3.6. Query an external table

This topic describes how to query an external table when a custom extractor is used. It also provides an example for reference.

In this example, the vehicle.csv file in the `/demo/SampleData/CustomTxt/AmbulanceData/` directory contains the following data:

```
1|1|51|1|46.81006|-92.08174|9/14/2017 0:00|S
1|2|13|1|46.81006|-92.08174|9/14/2017 0:00|NE
1|3|48|1|46.81006|-92.08174|9/14/2017 0:00|NE
1|4|30|1|46.81006|-92.08174|9/14/2017 0:00|W
1|5|47|1|46.81006|-92.08174|9/14/2017 0:00|S
1|6|9|1|46.81006|-92.08174|9/14/2017 0:00|S
1|7|53|1|46.81006|-92.08174|9/14/2017 0:00|N
1|8|63|1|46.81006|-92.08174|9/14/2017 0:00|SW
1|9|4|1|46.81006|-92.08174|9/14/2017 0:00|NE
1|10|31|1|46.81006|-92.08174|9/14/2017 0:00|N
```

Execute the following statement to submit a job, which calls a custom extractor to read data from OSS:

```
SELECT recordId, patientId, direction;
FROM ambulance_data_txt_external;
WHERE patientId > 25;
```

The following result is returned:

```
+-----+-----+-----+
| recordId | patientId | direction |
+-----+-----+-----+
| 1 | 51 | S |
| 3 | 48 | NE |
| 4 | 30 | W |
| 5 | 47 | S |
| 7 | 53 | N |
| 8 | 63 | SW |
| 10 | 31 | N |
+-----+-----+-----+
```

1.17.2.1.4. Use a custom extractor to read external unstructured data

This topic describes how to use a custom extractor to read external unstructured data. It also provides an example for reference.

The preceding topics describe how to use built-in and custom extractors to process text files such as CSV files that are stored in OSS. This topic describes how to use a custom extractor to process non-text files that are stored in OSS.

Audio files in the WAV format are used in this example.

1. Customize the `SpeechSentenceSnrExtractor` main logic. Call the `setup` method to read parameters, initialize the parameters, and import the audio processing model. You can use the `resource` function to import the model.

```
public SpeechSentenceSnrExtractor(){
    this.utteranceLabels = new HashMap<String, UtteranceLabel>();
}
@Override
public void setup(ExecutionContext ctx, InputStreamSet inputs,
    DataAttributes attributes){
    this.inputs = inputs;
    this.attributes = attributes;
    this.mlfFileName = this.attributes.getValueByKey(MLF_FILE_ATTRIBUTE_KEY);
    String sampleRateInKHzStr =
    this.attributes.getValueByKey(SPEECH_SAMPLE_RATE_KEY);
    this.sampleRateInKHz = Double.parseDouble(sampleRateInKHzStr);
    try {
        // read the speech model file from resource and load the model into
        memory
        BufferedInputStream inputStream =
        ctx.readResourceFileAsStream(mlfFileName);
        loadMlfLabelsFromResource(inputStream);
        inputStream.close();
    } catch (IOException e) {
        throw new RuntimeException("reading model from mlf failed with exception
```

```

"+ e.getMessage());
}
}
@Override
public Record extract() throws IOException {
    SourceInputStream inputStream = inputs.next();
    if (inputStream == null){
        return null;
    }
    // process one wav file to extract one output record [snr, id]
    String fileName = inputStream.getFileName();
    fileName = fileName.substring(fileName.lastIndexOf('/') + 1);
    logger.info("Processing wav file " + fileName);
    // infer id from speech file name
    String id = fileName.substring(0, fileName.lastIndexOf('.'));
    // read speech file into memory buffer
    long fileSize = inputStream.getFileSize();
    byte[] buffer = new byte[(int)fileSize];
    int readSize = inputStream.readToEnd(buffer);
    inputStream.close();
    // compute the avg sentence snr from speech file
    double snr = computeSnr(id, buffer, readSize);
    // construct output record [snr, id]
    Column[] outputColumns = this.attributes.getRecordColumns();
    ArrayRecord record = new ArrayRecord(outputColumns);
    record.setDouble(0, snr);
    record.setString(1, id);
    return record;
}
private void loadMlfLabelsFromResource(BufferedInputStream fileInputStream)
throws IOException {
    // loading MLF label from resource, skipped here
}
// compute the snr of the speech sentence, assuming the input buffer
contains the entire content of a wav file
private double computeSnr(String id, byte[] buffer, int validBufferLen){
    // computing the snr value for the wav file (supplied as byte buffer
array), skipped here
}

```

 **Note** The `extractor()` method implements the reading and processing logic of audio files. It calculates the signal-to-noise ratio (SNR) of the read data based on the audio processing model and writes the results to a record in the `[snr, id]` format.

2. Run the following commands to create an external table:

```
CREATE EXTERNAL TABLE IF NOT EXISTS speech_sentence_snr_external
(
  sentence_snr double,
  id string
)
STORED BY 'com.aliyun.odps.udf.example.speech.SpeechStorageHandler'
WITH SERDEPROPERTIES (
  'mLfFileName'='sm_random_5_utterance.text.label',
  'speechSampleRateInKHz' = '16'
)
LOCATION 'oss://<your-id>:<your-secret-key>@oss-cn-shanghai-internal.aliyuncs.com/oss-odps-test/dev/SpeechSentenceTest/'
USING 'odps-udf-example.jar,sm_random_5_utterance.text.label';
```

3. Run the following commands to read data from OSS:

```
SELECT sentence_snr, id
FROM speech_sentence_snr_external
WHERE sentence_snr > 10.0;
```

4. View the processing results:

```
-----
| sentence_snr | id |
-----
| 34.4703 | J310209090013_H02_K03_042 |
-----
| 31.3905 | tsh148_seg_2_3013_3_6_48_80bd359827e24dd7_0 |
-----
| 35.4774 | tsh148_seg_3013_1_31_11_9d7c87aef9f3e559_0 |
-----
| 16.0462 | tsh148_seg_3013_2_29_49_f4cb0990a6b4060c_0 |
-----
| 14.5568 | tsh_148_3013_5_13_47_3d5008d792408f81_0 |
-----
```

 **Note** A custom extractor allows you to execute SQL statements to process multiple audio files in OSS in a distributed manner. Similarly, you can use the large-scale computation capabilities of MaxCompute to process unstructured data, such as images and videos.

1.17.2.1.5. Data partitions

1.17.2.1.5.1. Overview

This topic describes the data partitioning feature of external tables.

LOCATION is used to specify an OSS directory, which is associated with an external table. MaxCompute reads all data in the OSS directory, including all files in the subdirectories. Due to a large amount of data accumulated in the directory, a full-text scan is performed. This operation results in extra I/O operations and prolongs the time required to process data. Two solutions are provided to address this issue:

- Reduce the amount of data: Plan data storage directories properly. Create multiple external tables for data from different parts, with LOCATION of each external table pointing to a subset of data.
- Partition data: Similar to internal tables, external tables support data partitioning. You can create partitions to manage the data.

Some examples are provided to demonstrate how to use the data partitioning feature of external tables.

1.17.2.1.5.2. Standard organization method and directory structure of partition data in OSS

This topic describes the standard organization method and directory structure of partition data in Object Storage Service (OSS). It also provides an example for reference.

Unlike the data in the internal tables of MaxCompute, the data stored in external storage, such as OSS, cannot be managed in MaxCompute. If you want to use the partitioned table feature of MaxCompute, make sure that the directories of data files in OSS are in the following format:

```
partitionKey1=value1\partitionKey2=value2\...
```

Example:

1. Your daily log files are stored in OSS, and you want to access some of the data from MaxCompute on a daily basis. If the log files are in the CVS format or a similar custom format, you can execute the following statement to create a partitioned external table:

```
CREATE EXTERNAL TABLE log_table_external (
  click STRING,
  ip STRING,
  url STRING,
)
PARTITIONED BY (
  year STRING,
  month STRING,
  day STRING
)
STORED BY 'com.aliyun.odps.CsvStorageHandler'
LOCATION 'oss://<ak_id>:<ak_key>@oss-cn-shanghai-internal.aliyuncs.com/oss-odps-test/log_data/';
```

 **Note** In the preceding example, the PARTITIONED BY clause is used to specify a partitioned external table. The partition keys are year, month, and day.

2. To make the partitions take effect, you must specify the OSS storage directory in the format shown in the preceding example. The following example shows a valid directory layout:

```
osscommand ls oss://oss-odps-test/log_data/
2017-09-14 08:03:35 128MB Standard oss://oss-odps-
test/log_data/year=2017/month=06/day=01/logfile
2017-09-14 08:04:12 127MB Standard oss://oss-odps-
test/log_data/year=2017/month=06/day=01/logfile.1
2017-09-14 08:05:02 118MB Standard oss://oss-odps-
test/log_data/year=2017/month=06/day=02/logfile
2017-09-14 08:06:45 123MB Standard oss://oss-odps-
test/log_data/year=2017/month=07/day=10/logfile
2017-09-14 08:07:11 115MB Standard oss://oss-odps-
test/log_data/year=2017/month=08/day=08/logfile
...
```

Note If you prepare offline data, use a tool to upload the offline data to OSS. In this case, you can specify the data directory format. For the partitioned table feature of the external table to work properly, we recommend that you use the directory format in the preceding example for the uploaded data.

3. You can execute the ALTER TABLE ADD PARTITION statement to import the partition information to MaxCompute. The following example shows sample statements:

```
ALTER TABLE log_table_external ADD PARTITION (year = '2017', month = '06', day = '01')
ALTER TABLE log_table_external ADD PARTITION (year = '2017', month = '06', day = '02')
ALTER TABLE log_table_external ADD PARTITION (year = '2017', month = '07', day = '10')
ALTER TABLE log_table_external ADD PARTITION (year = '2017', month = '08', day = '08')
...
```

4. If the data is ready and the partition information is imported to MaxCompute, you can execute SQL statements to manage the partitions in the external table in OSS.
 - o Execute the following statement to count the number of unique IP addresses in the log generated on June 1, 2017:

```
SELECT count(distinct(ip)) FROM log_table_external;
WHERE year = '2017' AND month = '06' AND day = '01';
```

Note In this example, MaxCompute accesses only logfile and logfile.1 in the *log_data/year=2016/month=06/day=01* sub-folder but not all files in the *log_data* folder for the external table *log_table_external*. This prevents unnecessary I/O operations.

- o Similarly, you can execute the following statement to analyze data from the second half of year 2017:

```
SELECT count(distinct(ip)) FROM log_table_external;
WHERE year = '2017' AND month > '06';
```

Note In this example, only the logs for the second half of year 2017 stored in OSS are accessed.

1.17.2.1.5.3. Custom directories of partition data in OSS

This topic describes the custom directories in which partition data is saved in Object Storage Service (OSS). It also provides an example for reference.

If you have historical data stored in OSS but the data is not saved in the *partitionKey1=value1\partitionKey2=value2\...* format, you can still access the data by using the partitioning feature of MaxCompute. MaxCompute provides a way to import partitions from custom directories.

Example:

1. The directories in which partition data is saved contain only partition values but do not contain partition keys. The following example shows the layout of the data directories:

```
osscmd ls oss://oss-odps-test/log_data_customized/
2017-09-14 08:03:35 128MB Standard oss://oss-odps-
test/log_data_customized/2017/06/01/logfile
2017-09-14 08:04:12 127MB Standard oss://oss-odps-
test/log_data_customized/2017/06/01/logfile.1
2017-09-14 08:05:02 118MB Standard oss://oss-odps-
test/log_data_customized/2017/06/02/logfile
2017-09-14 08:06:45 123MB Standard oss://oss-odps-
test/log_data_customized/2017/07/10/logfile
2017-09-14 08:07:11 115MB Standard oss://oss-odps-
test/log_data_customized/2017/08/08/logfile
...
```

2. You can run the following command to bind subdirectories to different partitions:

```
ALTER TABLE log_table_external ADD PARTITION (year = '2017', month = '06', day = '01')
LOCATION 'oss://<ak_id>:<ak_key>@oss-cn-shanghai-internal.aliyuncs.com/oss-odps-test/log_data_customiz
ed/2017/06/01/';
```

 **Note** The LOCATION parameter specifies an OSS directory. This directory maps to the partition that is added to the external table by using ADD PARTITION. This way, MaxCompute can access data in the OSS directory even if the directory is not in the partitionKey1=value1\partitionKey2=value2\...format. In addition, you can access the partition data in the subdirectory.

1.17.2.1.5.4. Access fully-customized non-partitioned data subsets

This topic describes how to access fully-customized non-partitioned data subsets.

In some special cases, you may need to access an arbitrary subset of a file in an OSS directory, and the format of the directory where the files in this file subset are saved is not deterministic. The unstructured data processing framework of MaxCompute is able to handle this situation, but is not discussed in this topic.

For more information about advanced operations, contact the MaxCompute technical team.

1.17.2.1.6. Output OSS data

1.17.2.1.6.1. Create an external table

This topic describes how to create an external table when you write data to OSS. It also provides an example for reference.

To write data to OSS, you must execute the CREATE EXTERNAL TABLE statement to create an external table first. The process is the same as that of reading data from OSS. After the external table is created, you can execute MaxCompute SQL statements such as INSERT INTO or OVERWRITE. The following example demonstrates how to create an external table by using the built-in storage handler TsvStorageHandler of MaxCompute.

```
DROP TABLE IF EXISTS tpch_lineitem_tsv_external;  
CREATE EXTERNAL TABLE IF NOT EXISTS tpch_lineitem_tsv_external  
(  
  orderkey BIGINT,  
  supkey BIGINT,  
  discount DOUBLE,  
  tax DOUBLE,  
  shipdate STRING,  
  linestatus STRING,  
  shipmode STRING,  
  comment STRING  
)  
STORED BY 'com.aliyun.odps.TsvStorageHandler'  
LOCATION 'oss://<AK_id>:<AK_secret>@oss-cn-hangzhou-zmf.aliyuncs.com/oss-odps-test/tsv_output_folder/';
```

Note

The preceding DDL statement creates an external table named `tpch_lineitem_tsv_external`, and associates two external data dimensions with this external table.

- Data storage medium: `LOCATION` associates an OSS address with the external table. This address is used to read data from or write data to the external table.
- Data storage format: A storage handler is used to define the data access mode. In this example, the built-in storage handler `com.aliyun.odps.TsvStorageHandler` of MaxCompute is used to read or write data from or to TSV files. You can also use the MaxCompute SDK to define storage handlers.

1.17.2.1.6.2. Write data to a TSV text file by using an INSERT statement on an external table

This topic describes how to write data to a TSV text file by using an `INSERT` statement on an external table. It also provides an example for reference.

After you associate an OSS object with an external table, you can execute a standard `INSERT OVERWRITE` or `INSERT INTO` statement on the external table to write data to the OSS file. The data source can be either the data stored in MaxCompute internal tables or the external data that is imported into MaxCompute by using an external table.

Note

- MaxCompute internal table: You can execute an `INSERT` statement on an external table to write data from a MaxCompute internal table to an external storage medium.
- External data that is imported into MaxCompute by using an external table: You can import external data to MaxCompute by using an external table, use the data for computations, and then export the results to an external storage directory or external data store. For example, you can import Tablestore data into MaxCompute and then export the data to OSS.

In this example, you have a MaxCompute internal table named `tpch_lineitem` and want to export some of the data in the table to OSS in the TSV format. After you create an external table, you can execute the `INSERT OVERWRITE` statement to export the data:

```
INSERT OVERWRITE TABLE tpch_lineitem_tsv_external;
SELECT l_orderkey, l_suppkey, l_discount, l_tax, l_shipdate, l_linestatus,
l_shipmode, l_comment
FROM tpch_lineitem
WHERE l_discount = 0.07 and l_tax = 0.01;
```

In the preceding example, eight columns are selected from the rows that meet the conditions `l_discount = 0.07` and `l_tax = 0.01` in the internal table `tpch_lineitem` and then written to OSS in the TSV format. The selected columns in the internal table correspond to the schema of the external table `tpch_lineitem_tsv_external`. After this operation is complete, you can view the related TSV data file in OSS.

Notice

Data exported from MaxCompute to OSS is stored in a special file structure.

- If you use MaxCompute to execute the `INSERT INTO` or `INSERT OVERWRITE` statement on an external table and write data to an OSS directory, all data is saved in a `.odps` folder in the directory specified by `LOCATION`.
- The `.meta` file in the `.odps` folder is an extra macro data file written by MaxCompute. This file records valid data in the folder. If the `INSERT` operation succeeds, all data in the folder is valid. You are only required to parse the macro data if a job fails.
- If a job fails or is terminated, execute the `INSERT OVERWRITE` statement again until it is complete. This prevents the `.meta` file from being parsed.
- If you want to parse the `.meta` file, contact Alibaba Cloud technical team.

The number of files that are generated during the internal processing of TSV and CSV files of MaxCompute is equal to the number of concurrent SQL stages. You can use the flexible semantics and configurations of MaxCompute to limit the number of generated files. In the preceding example, if you want to force the generation of a TSV file, you can append `DISTRIBUTE BY l_discount` to the `INSERT OVERWRITE` statement. Then, a reduce stage with only one reducer is added at the end so that only one TSV file is generated.

1.17.2.1.6.3. Write data to an unstructured file by using an INSERT statement on an external table

This topic describes how to write data to an unstructured file by using an `INSERT` statement on an external table.

MaxCompute provides the Outputter interface for data output. You can call this interface to write user data to a custom unstructured data file by using `OutputStream`. Further details are not provided in this topic.

If you have related requirements, contact the MaxCompute technical team.

1.17.2.1.6.4. Migrate data between different storage media by using MaxCompute

This topic describes how to migrate data between different storage media by using MaxCompute. It also provides an example for reference.

External tables act as an interface between MaxCompute and external storage media. External tables can be used to read data from or write data to various external storage media. Based on the feature of external tables, various links can be established to compute and store data. For example:

- MaxCompute reads data from an Object Storage Service (OSS) bucket that is associated with External table

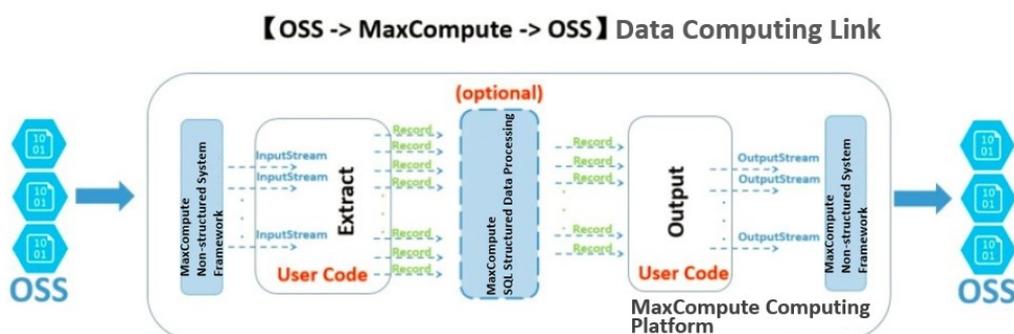
A, performs complex computations, and then generates results to the directory of the OSS bucket that is associated with External table B.

- MaxCompute reads data from External table A, performs complex computations, and then generates results to the directory of the OSS bucket that is associated with External table B.

Note In the preceding examples, the data source for the SELECT statement is an external table, rather than a MaxCompute internal table.

The following figure shows the flowchart for data migration.

MaxCompute is used as a central computing platform. It reads data from an OSS bucket and then writes the data to another OSS bucket (in a different location or within a different OSS account).



Based on the data flow and processing logic shown in the preceding figure, the unstructured data processing framework can be considered a coupled data ingress and egress at both ends of MaxCompute.

1. The external data from an OSS bucket is converted based on the unstructured framework, and provided to the InputStream class of Java. The extract logic is only used to read, parse, transform, and compute the data from the InputStream class, and return the data in the Record format used by MaxCompute.
2. Part of the returned records are used in the SQL logical operations on MaxCompute. These operations utilize the powerful SQL computation engine that is built in MaxCompute, and may generate new records.
3. The records obtained after computations are transferred to your custom output logic for further computations. Finally, the required information is extracted from the records, transferred by using OutputStream, and written to OSS.

Note You can perform the preceding steps based on your business requirements.

1.17.2.1.7. STS mode authorization for OSS

This topic describes the Security Token Service (STS) mode authorization for Object Storage Service (OSS), and provides an example for reference.

When you create an external table, the Location-based OSS access account allows you to enter AccessKey ID and AccessKey secret in plaintext mode. However, the account information may be exposed. To prevent account information from being exposed, MaxCompute provides a more secure way to access OSS.

MaxCompute integrates RAM and STS of Alibaba Cloud to enhance account security. You can use one of the following methods to grant permissions:

- If the owners of MaxCompute and OSS use the same Alibaba Cloud account, you can authorize access to OSS with one click in the RAM console.

- Custom authorization is supported.
 - i. You can log on to the RAM console and authorize access to OSS.

Create a role named `AliyunODPSDefaultRole` or `AliyunODPSRoleForOtherUser` and compile the following policy content:

```
-- If the owners of MaxCompute and OSS use the same Alibaba Cloud account:
{
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "odps.aliyuncs.com"
        ]
      }
    },
    {
      "Action": "sts:AssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "ID of the Alibaba Cloud account that owns the MaxCompute project@odps.aliyuncs.com"
        ]
      }
    }
  ],
  "Version": "1"
}

-- If the owners of MaxCompute and OSS use different Alibaba Cloud accounts:
{
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "ID of the Alibaba Cloud account that owns the MaxCompute project@odps.aliyuncs.com"
        ]
      }
    }
  ],
  "Version": "1"
}
```

- ii. Grant the `AliyunODPSRolePolicy` permission, which is required for the role to access OSS.

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "oss:ListBuckets",
        "oss:GetObject",
        "oss:ListObjects",
        "oss:PutObject",
        "oss:DeleteObject",
        "oss:AbortMultipartUpload",
        "oss:ListParts"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

-- You can also add other permissions as required.

- iii. Grant the AliyunODPSRolePolicy permission to the role.

 **Note** After authorization is complete, view the role details to obtain the RAM information of this role. You must specify the RAM information when you create an OSS external table later.

1.17.2.2. Tablestore data source

1.17.2.2.1. Overview

This topic describes how to access and process data of Tablestore data sources in MaxCompute.

As the core computing component of the Alibaba Cloud big data platform, MaxCompute meets most distributed computing requirements within and outside Alibaba Group. MaxCompute SQL provides powerful support for the quick processing and storage of exabytes of offline data. As big data businesses continue to grow, many new data usage scenarios emerge. To adapt to the new scenarios, the MaxCompute computing framework is constantly evolving. Its powerful computation capabilities, originally designed to process internal data in special formats, have expanded to process external data sources in various formats. This topic describes how to import data from Tablestore to MaxCompute, which implements seamless interoperability between data sources.

In online service scenarios, NoSQL KV storage (such as Bigtable or HBase) features flexible schema, easy scalability, and high real-time performance, compared with traditional databases. Alibaba Cloud Tablestore is a large-scale NoSQL data storage service based on the Apsara distributed operating system. It stores and allows real-time access to large volumes of key-value pairs. Tablestore is widely used by all business units in Alibaba Group and the Alibaba Cloud ecosystem. In particular, Tablestore features, such as row-level real-time update and override writing, supplement the append-only operations of MaxCompute tables. As a storage-oriented service, Tablestore does not provide sufficient computing capabilities to concurrently process large volumes of data. In this case, MaxCompute allows you to create external tables to access Tablestore data sources.

Examples are provided to demonstrate how MaxCompute accesses and processes Tablestore data.

 **Note** This topic assumes that you have a basic knowledge of Tablestore operations. If you are not familiar with Tablestore or are new to the concept of key-value tables, you can first learn some basic Tablestore concepts, such as primary keys, partition keys, and attribute columns.

1.17.2.2.2. MaxCompute reads and computes data in Table Store

1.17.2.2.2.1. Create an external table

This topic describes how to create an external table for Tablestore data sources. It also provides an example for reference.

MaxCompute accesses Tablestore data by using external tables. After you execute the `CREATE EXTERNAL TABLE` statement to introduce the description of Tablestore table data to the metadata of MaxCompute, you can process Tablestore data in the same way as you process a standard table.

Execute the following statement to create an external table:

```
DROP TABLE IF EXISTS ots_table_external;
CREATE EXTERNAL TABLE IF NOT EXISTS ots_table_external
(
  odps_orderkey bigint,
  odps_orderdate string,
  odps_custkey bigint,
  odps_orderstatus string,
  odps_totalprice double
)
STORED BY 'com.aliyun.odps.TableStoreStorageHandler'
-- com.aliyun.odps.TableStoreStorageHandler is a MaxCompute built-in storage handler that processes Tablestore
data. It defines the interaction between MaxCompute and Tablestore. The relevant logic is implemented by MaxCo
mpute.
WITH SERDEPROPERTIES (
  -- SERDEPROPERTIES is considered an interface that provides parameter options. Two options must be specified fo
r TableStoreStorageHandler: tablestore.columns.mapping and tablestore.table.name.
  'tablestore.columns.mapping'=':o_orderkey,:o_orderdate,o_custkey,
  o_orderstatus,o_totalprice',
  -- tablestore.columns.mapping: This option is required. It describes the columns of Tablestore tables that are acces
sed by MaxCompute. The columns include primary key columns and attribute columns. Columns whose names star
t with a colon (:) are primary key columns in Tablestore tables. In this example, :o_orderkey and :o_orderdate are pr
imary key columns. Other columns are attribute columns. A Tablestore table can have one to four primary keys of t
he BIGINT or STRING type. The first primary key is the partition key. If you specify a table mapping, you must provid
e all primary key columns in the specified Tablestore table. You can specify only the attribute columns that are acces
sed by MaxCompute.
  'tablestore.table.name'='ots_tpch_orders'
  -- tablestore.table.name: This option is required. It describes the names of Tablestore tables that are accessed by M
axCompute. If you specify an invalid (nonexistent) Tablestore table name, an error is returned and MaxCompute do
es not create a Tablestore table with this name.
)
LOCATION 'tablestore://<your AK id*>:<your AK secret key*>@odps-ots-dev.cn-
hangzhou.ots.aliyuncs.com';
-- The LOCATION clause specifies the Tablestore information, including the instance name and endpoint.
```

 **Note** The preceding example maps a Tablestore table to an external table of MaxCompute. After the mapping is specified, subsequent operations on Tablestore data can be performed by using the external table.

1.17.2.2.2. Use an external table to access Tablestore data

This topic describes how to use an external table to access TableStore data and perform computations.

After you create an external table, Tablestore data is introduced into the MaxCompute ecosystem. Then, you can access Tablestore data by using the MaxCompute SQL syntax.

Example:

```
SELECT odps_orderkey, odps_orderdate, SUM(odps_totalprice) AS totalprice
FROM ots_table_external
WHERE odps_orderkey > 5000 AND odps_orderdate > 20170725 AND odps_orderdate < 20170910
GROUP BY odps_orderkey, odps_orderdate
HAVING totalprice > 2000;
```

 **Note** In this example, the MaxCompute SQL syntax that you are familiar with is directly used, and all the other implementations of accessing Tablestore data are performed by MaxCompute.

If you want to compute one piece of data multiple times, you can import the data from Tablestore into an internal table of MaxCompute. This way, you do not need to read the data from Tablestore each time you compute the data.

Example:

```
CREATE TABLE internal_orders AS
SELECT odps_orderkey, odps_orderdate, odps_custkey, odps_totalprice
FROM ots_table_external
WHERE odps_orderkey > 5000 ;
```

 **Note** In this example, `internal_orders` is a MaxCompute table that you are familiar with. This table has all features of MaxCompute internal tables, such as efficient column compression for data storage and complete metadata. This table is stored in MaxCompute. Therefore, it can be accessed faster than an external table in Tablestore. This feature is particularly suitable for hotspot data that is used for multiple computations.

1.17.2.2.3. Write data from MaxCompute to Tablestore

This topic describes how to use external tables to write data from MaxCompute to Tablestore.

Data interaction between MaxCompute and Tablestore includes importing data from Tablestore to MaxCompute for batch processing and exporting the data processing results from MaxCompute to Tablestore. Tablestore features, such as real-time update and single-row overwriting, allow you to quickly upload offline computing results to online applications. The `INSERT OVERWRITE` statement is used to export data processing results from MaxCompute to Tablestore.

 **Note** MaxCompute does not proactively create external tables in Tablestore. Before you export data to a table in Tablestore, make sure that the table exists in Tablestore. If the table does not exist, an error is returned.

For example, you have created the external table `ots_table_external` in MaxCompute to access data of the `ots_tpch_orders` table in Tablestore. A data record named `internal_orders` is stored in MaxCompute. To process the data record `internal_orders` and write the processing results to Tablestore, execute the `INSERT OVERWRITE TABLE` statement. Sample statement:

```
INSERT OVERWRITE TABLE ots_table_external
SELECT odps_orderkey, odps_orderdate, odps_custkey, CONCAT(odps_custkey,
'SHIPPED'), CEIL(odps_totalprice)
FROM internal_orders;
```

Note

- Tablestore is a NoSQL service that stores data in the format of key-value pairs. Data outputs from MaxCompute affect only the rows that contain the primary keys of the Tablestore table. In addition, only the attribute columns specified when you create the table are updated. The columns that are not included in the external table are not modified.
- If you execute the `INSERT OVERWRITE` statement on the external table, MaxCompute inserts 200 data records into the table at a time by default. You can adjust a batch size to limit the total batch size to 4 MB.

1.17.2.3. AnalyticDB data source

1.17.2.3.1. Overview

This topic describes how to access and process data of AnalyticDB for MySQL data sources in MaxCompute.

AnalyticDB for MySQL updates or processes data. If both the data processed by AnalyticDB for MySQL and the data in MaxCompute are used for data computations, the data from AnalyticDB for MySQL must be synchronized with the data from MaxCompute. Therefore, you must create an external table to access the data of AnalyticDB for MySQL data sources.

The following example shows how MaxCompute accesses and processes the data of AnalyticDB for MySQL data sources.

1.17.2.3.2. Write data to AnalyticDB

1.17.2.3.2.1. Create an external table

This topic describes how to create external tables for AnalyticDB for MySQL data sources and provides an example for reference.

Run the following command to create an external table:

```
set odps.sql.hive.compatible=true;
drop table if exists ads_table_external;
CREATE EXTERNAL TABLE if not exists ads_table_external
(
  c_int int,
  c_tinyint tinyint,
  c_boolean boolean,
  c_smallint smallint,
  c_bigint bigint,
  c_double double,
  c_float float,
  --c_time datetime,
  c_date date,
  c_timestamp datetime,
  c_string string
)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://host:port/databasename?useSSL=false&user=${user}&password=${password}&table=${tablename}'
```

1.17.2.3.2.2. Write and query data

This topic describes how to use external tables to write and query AnalyticDB for MySQL data.

After an external table is created, you can use it the same way that you use a standard table. You can use the INSERT OVERWRITE, INSERT INTO, and SELECT statements to write data to the external table and check whether the write operation succeeds. For more information about the usage of these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.2.3.3. Read data from AnalyticDB for MySQL

This topic describes how to read data from AnalyticDB for MySQL and provides an example for reference.

If you have compiled the ads_read_external script, execute the following statements to read data from AnalyticDB for MySQL:

```

set odps.sql.hive.compatible=true;
drop table if exists ads_read_external;
CREATE EXTERNAL TABLE if not exists ads_read_external
(
  c_int int ,
  c_tinyint tinyint ,
  c_boolean boolean ,
  c_smallint smallint ,
  c_bigint bigint ,
  c_double double ,
  c_float float ,
  --c_time datetime ,
  c_date date ,
  c_timestamp datetime ,
  c_string string
)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://host:port/databasename?useSSL=false&user=${user}&password=${password}&table=${tablename}'
-- Create an external table.
select * from ads_read_external;
-- Query and read data.

```

1.17.2.4. RDS data source

1.17.2.4.1. Overview

This topic describes how to access and process data of ApsaraDB RDS data sources in MaxCompute.

ApsaraDB RDS updates or processes data. If both the data processed by ApsaraDB RDS and the data in MaxCompute are used for computations, the data from ApsaraDB RDS must be synchronized with the data from MaxCompute. Therefore, you must create an external table to access the ApsaraDB RDS data source.

Some examples are provided to demonstrate how MaxCompute accesses and processes data of ApsaraDB RDS data sources.

 **Note** When you create an external table, you are not required to create the related table in ApsaraDB RDS. However, when you perform the SELECT or INSERT operation on external tables, you must first create the related tables in ApsaraDB RDS.

1.17.2.4.2. Write data to RDS

1.17.2.4.2.1. Create an external table

This topic describes how to create an external table for ApsaraDB RDS data sources. It also provides an example for reference.

Execute the following statements to create an external table:

 **Note** ApsaraDB RDS for MySQL versions earlier than V8.0 are supported. If the ApsaraDB RDS for MySQL version is 8.0 or later, the error `FAILED: Generating job conf failed, gen jobconf failed` is returned. In this case, the external table cannot be created.

```
set odps.sql.hive.compatible=true;
drop table if exists rds_table_external;
CREATE EXTERNAL TABLE if not exists rds_table_external
(
  id bigint,
  name string,
  age tinyint
)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://host:port/databasename?useSSL=false&user=${user}&password=${password}&table=${tablename}'
```

1.17.2.4.2.2. Write and query data

This topic describes how to use external tables to write and query ApsaraDB RDS data.

After an external table is created, you can use it the same way you use a common data table. You can execute the INSERT OVERWRITE, INSERT INTO, and SELECT statements to write data and check whether the write operation succeeds. For more information about the usage of these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.2.4.3. Read data from ApsaraDB RDS

This topic describes how to read data from ApsaraDB RDS and provides an example for reference.

If you have compiled the rds_read_external script, run the following commands to read data from ApsaraDB RDS:

```
set odps.sql.hive.compatible=true;
drop table if exists rds_read_external;
CREATE EXTERNAL TABLE if not exists rds_read_external
(
  id int,
  name string,
  age int
)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://host:port/databasename?useSSL=false&user=${user}&password=${password}&table=${tablename}'
-- Create an external table.
select * from rds_read_external;
-- Query and read data from the external table.
```

1.17.2.5. HDFS data source (Alibaba Cloud)

1.17.2.5.1. Overview

This topic describes how to access and process Apsara File Storage for HDFS data sources in MaxCompute.

Apsara File Storage for HDFS is a distributed file system designed for Alibaba Cloud computing resources such as Elastic Compute Service (ECS) and Container Service.

Apsara File Storage for HDFS allows you to manage and access data in the same way as the open source Hadoop Distributed File System (HDFS). You can use a distributed file system without the need to modify existing big data applications. The distributed file system offers various features such as unlimited capacity, performance expansion, single namespace, multi-party sharing, high reliability, and high availability.

MaxCompute can interact with Apsara File Storage for HDFS for joint computations after you create external tables.

Apsara File Storage for HDFS supports multiple file formats, such as text file, sequence file, RC file, Parquet, and AVRO. The following example shows how MaxCompute accesses and processes data of Apsara File Storage for HDFS. The text file format is used in this example.

1.17.2.5.2. Data processing for common tables

1.17.2.5.2.1. Write data to HDFS

Create an external table

This topic describes how to create an external table for an Apsara File Storage for HDFS data source. It also provides an example for reference.

Run the following commands to create an external table after the `textfile_external` script is compiled:

```
set odps.sql.hive.compatible=true;
drop table if exists textfile_external;
CREATE external TABLE if not exists textfile_external
(
  c_int int ,
  c_tinyint tinyint ,
  c_boolean boolean ,
  c_smallint smallint ,
  c_bigint bigint ,
  c_double double ,
  c_float float ,
  c_date date ,
  c_timestamp datetime ,
  c_string string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES ('field.delim','=',)
STORED AS textfile
location "mcfed:dfs://host:port/user/textfile"
-- host must be set to MountPointId.
-- /user/textfile is the file directory. Replace it with the actual file directory.
TBLPROPERTIES(
  "mcfed.fs.dfs.impl"="com.alibaba.dfs.DistributedFileSystem"
);
```

Write and query data

This topic describes how to use external tables to write and query data of Apsara File Storage for HDFS.

After an external table is created, you can use it the same way you use a common data table. You can use `INSERT OVERWRITE`, `INSERT INTO`, and `SELECT` statements to write data and check whether the write operation succeeds. For more information about the usage of these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.2.5.2.2. Read data from Apsara File Storage for HDFS

This topic describes how to read data from Apsara File Storage for HDFS and provides an example for reference.

Run the following commands to read data from Apsara File Storage for HDFS after you compile the `textfile_external_read` script:

```
set odps.sql.hive.compatible=true;
drop table if exists textfile_external_read;
CREATE external TABLE if not exists textfile_external_read
(
  c_int int ,
  c_tinyint tinyint ,
  c_boolean boolean ,
  c_smallint smallint ,
  c_bigint bigint ,
  c_double double ,
  c_float float ,
  c_date date ,
  c_timestamp datetime ,
  c_string string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES ('field.delim=',)
STORED AS textfile
location "mcfed:dfs://host:port/user/textfile"
-- host must be set to MountPointId.
-- /user/textfile is the file directory. Replace it with the actual file directory.
TBLPROPERTIES(
  "mcfed.fs.dfs.impl"="com.alibaba.dfs.DistributedFileSystem"
);
-- Create an external table.
select * from textfile_external_read;
select count(*) from textfile_external_read;
select a.c_int,a.c_boolean,a.c_string,b.value from textfile_external_read a join dfstest b on a.c_int=b.id;
-- Query and read data from the external table.
```

1.17.2.5.3. Data processing for partitioned tables

This topic describes how to use external tables to process data of open source Hadoop Distributed File System (HDFS) data sources in partitioned tables. It also provides an example for reference.

Execute the following statements to create an external table and add partitions to the table to process data:

```

set odps.sql.hive.compatible=true;
drop table if exists textfile_partition;
CREATE external TABLE if not exists textfile_partition
(
  id string,
  name string
)
partitioned by (date string)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES ('field.delim'=',')
STORED AS textfile
location "mcfed:dfs://host:port/user/partition/textfile/"
-- host in this command indicates MountPointId.
-- user/partition/textfile/ is the file directory. Replace it with the actual file directory.
TBLPROPERTIES(
  "mcfed.fs.dfs.impl"="com.alibaba.dfs.DistributedFileSystem"
);
-- Create an external table.
alter table textfile_partition add partition (date='20190218');
alter table textfile_partition add partition (date='20190219');
-- Add partitions to the external table.
insert into table textfile_partition partition(date='20190218')select '1','cd' from (select count(*) from textfile_partition)a;
insert into table textfile_partition partition(date='20190219')select '2','gh' from (select count(*) from textfile_partition)a;
-- Write data to HDFS.
select * from textfile_partition;
select count(*) from textfile_partition;
select a.id,a.name,b.value from textfile_partition a join dfstest b on a.id=b.id;
-- Query and read data from the external table.

```

1.17.2.6. TDDL data source

1.17.2.6.1. Overview

This topic describes how to access and process data of Taobao Distributed Data Layer (TDDL) data sources in MaxCompute.

TDDL is a database middleware that is widely used within Alibaba Group. It supports data sharding, read and write splitting, and failovers by encapsulating MySQL databases. In most cases, TDDL can be directly used to access MySQL databases. TDDL also provides the Corona connection mode. Corona is a MySQL proxy that complies with the standard MySQL protocol and can use a Java Database Connectivity (JDBC) driver to establish a connection.

MaxCompute can access MySQL databases of TDDL. Built-in storage handlers encapsulate native APIs provided by Hadoop such as org, apache, hadoop, mapreduce, lib, and db. The JDBC driver of the MySQL database is used for data communication at the underlying layer.

Some examples are provided to demonstrate how MaxCompute accesses and processes data of TDDL data sources.

 **Notice** In terms of create, read, update, and delete (CRUD) operations, only the following operations are supported:

- MaxCompute reads data from the external table that is created for a MySQL database.
- MaxCompute writes data to the external table in append mode.

1.17.2.6.2. Preparations

This topic describes the preparations that are required before you create an external table to process data of a Taobao Distributed Data Layer (TDDL) data source.

By default, many features are disabled in MaxCompute V2.0. Therefore, you must manually configure the following parameters to use a new framework to process unstructured data:

```
set odps.sql.hive.compatible=true;
-- Set this parameter for all DDL and DML statements that are used on the external tables for TDDL.
```

```
set odps.sql.udf.java.retain.legacy=false;
-- Set this parameter for all DDL and DML statements that are used on the external tables for TDDL.
```

```
set odps.sql.jdbc.splits.num=3;
-- Set the number of instance splits that are used by MaxCompute to read data from the MySQL database. Maximum value: 256. Default value: 1. You must set this parameter for the SELECT operation on external tables for TDDL.
```

```
set odps.sql.jdbc.reducer.num=3;
-- Set the number of concurrent instances that are used by MaxCompute to write data to the MySQL database. Maximum value: 256. Default value: 64. If the number of concurrent instances in the generated execution plan is less than this value, you do not need to change this value. You must set this parameter if you need to perform the INSERT operation on external tables for TDDL.
```

```
set odps.sql.hive.compatible=true;
-- Call an API operation defined by the open source community to obtain and parse the data types of the MySQL database. You must set this parameter for all DDL and DML statements that are used on the external tables for TDDL.
```

```
set odps.sql.type.system.odps2=true;
-- Set this parameter if new data types are involved in SQL statements, such as CREATE, SELECT, and INSERT. The new data types are TINYINT, SMALLINT, INT, FLOAT, VARCHAR, TIMESTAMP, and BINARY.
```

1.17.2.6.3. Create an external table

1.17.2.6.3.1. Syntax

This topic describes the syntax that is used to create an external table for a Taobao Distributed Data Layer (TDDL) data source.

External tables can be used as interfaces between MaxCompute and databases. The method that is used to process unstructured data of MySQL databases in TDDL is similar to the method used to access and process OSS unstructured data. To create an external table, you must execute the CREATE EXTERNAL TABLE statement first. The following example shows the syntax:

```

DROP TABLE [IF EXISTS] <external_table_name>;
CREATE EXTERNAL TABLE [IF NOT EXISTS] <external_table_name>
(<column schemas>)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://path_format'
TBLPROPERTIES(
...
);

```

Parameters:

- column schema: supports the data types that are listed in the following table. Data type mapping

MySQL Type Name	MaxCompute Type Name
TINYINT(UNSIGNED)	TINYINT
SMALLINT(UNSIGNED)	SMALLINT
INT(UNSIGNED)	INT
BIGINT(UNSIGNED)	BIGINT
BOOLEAN	BOOLEAN
FLOAT	FLOAT
DOUBLE	DOUBLE
VARCHAR	VARCHAR
TEXT	STRING
DATE	DATE
DATETIME	DATETIME
DECIMAL	DECIMAL(x, y) (The default precision is (10, 0). An error is returned when an overflow occurs.)

 **Notice** UNSIGNED data types are not supported in MaxCompute. Therefore, loss of precision may occur if UNSIGNED data types are specified. Handle this issue based on scenarios:

- `setproject odps.sql.udf.strict.mode=true;` (strict mode, which is the default mode)
 - Read data from an external table: If loss of precision does not occur when data of the UNSIGNED data type is converted into the SIGNED data type, MaxCompute reads and converts data normally. If loss of precision occurs during data type conversion, the RuntimeException ("value out of range") error is returned.
 - Write data into an external table: MaxCompute does not check data types. You can enable the MySQL database to produce the expected behavior by setting SQL_Mode of the MySQL database. For more information about the setting and behavior of SQL_Mode, see [Server SQL Modes](#).
- `setproject odps.sql.udf.strict.mode=false;` (non-strict mode, which needs to be explicitly configured)
 - Read data from an external table: If loss of precision does not occur when data of the UNSIGNED data type is converted into the SIGNED data type, MaxCompute reads and converts data normally. If loss of precision occurs during data type conversion, NULL is returned.
 - Write data to an external table: MaxCompute does not check data types. You can enable the MySQL database to produce the expected behavior by setting SQL_Mode of the MySQL database.

- **STORED BY:** Only built-in storage handlers are supported. Table field types of TDDL must be the same as the data types that are specified by column schema.
- **LOCATION:** The following three methods can be used to connect to a location:

Method 1: Connect to the MySQL database LOCATION by using a Java Database Connectivity (JDBC) connection string.

```
jdbc:mysql://<user>:<password>@<host>/<databaseName>?useSSL=false&table=<tableName>
```

user is the username of the JDBC connection string that is used to connect to the MySQL database.
password is the password of the JDBC connection string used to connect to the MySQL database. host is the network address of the MySQL database instance. databaseName is the name of the MySQL database.
tableName is the name of the MySQL database table that corresponds to the external table.

Method 2: Connect to the database LOCATION of TDDL in Corona mode.

```
jdbc:mysql://<user>:<password>@<host>/<databaseName>?useSSL=false&table=<tableName>
```

Method 3: Connect to the database LOCATION of TDDL based on appname.

```
jdbc:mysql://dummy_host?table=<tableName>
```

tableName is the name of the MySQL table that corresponds to the external table. You must specify `odps.federation.jdbc.tddl.appname` in the TBLPROPERTIES clause.

 Notice

In method 1, MaxCompute interacts with the database by using a direct JDBC connection. You must write the username and password in plaintext. This poses a security risk. Usernames and passwords are hidden when Logview or DESC EXTENDED TABLE is used in MaxCompute. For security concerns, we recommend that you use a separate DDL statement to create an external table before you use the external table.

For example, a project member with higher permissions can create an external table in MaxCompute. Then, other project members can directly use the external table. This prevents project members with lower permissions from using the plaintext username and password. In this case, you do not need to include the plaintext password in SQL scripts.

- TBLPROPERTIES: includes the following items.
 - odps.federation.jdbc.condition: specifies the filter condition that is used when MaxCompute reads data from the MySQL database. Difference between `odps.federation.jdbc.condition` and `select * from text_test_jdbc_write_external where condition` :

For example, the MySQL database table contains 100 rows of data and you want to obtain 10 data records that meet the specified condition. If you run `odps.federation.jdbc.condition`, data records of the MySQL database table is filtered and MaxCompute reads only 10 data records from the external table. If you run `select * from text_test_jdbc_write_external where condition`, MaxCompute reads 100 data records from the MySQL database table and then obtains 10 data records.
 - odps.federation.jdbc.colmapping: specifies the column name mapping. Example:

```
-- mysql schema: mysqlId int
-- MaxCompute create table
CREATE EXTERNAL TABLE if not exists table_external
(
  odpsId1 int,
  odpsId2 int
)
STORED BY ...
location ...
TBLPROPERTIES('odps.federation.jdbc.colmapping'='odpsId1:mysqlId, odpsId2:mysqlId');
```

- odps.federation.jdbc.insert.type: specifies the insertion type when data is written to the MySQL database. Only the following types are supported: SimpleInsert, InsertOnDuplicateKeyUpdate, and ReplaceInto. If you do not set this parameter, the default value is SimpleInsert.

The INSERT statement that is executed in MaxCompute is parsed into the following SQL statements to update data in the database:

```
insert into sqlTable xxx values xxx;
insert into sqlTable xxx values xxx on duplicate key update col1=values(col1), col2=values(col2);
replace into sqlTable xxx values xxx;
```
- odps.federation.jdbc.tddl.app.access.key: the AccessKey ID for the authorized application.
- odps.federation.jdbc.tddl.app.secret.key: the AccessKey secret for the authorized application.
- odps.federation.jdbc.tddl.appname: the application name of TDDL. Note that if you specify this parameter, MaxCompute uses the application name to access the MySQL database by using the TDDL SDK.

1.17.2.6.3.2. Example

This topic provides an example of how to create an external table for a Taobao Distributed Data Layer (TDDL) data source.

The following example shows how to connect to the database of TDDL based on appname. In this example, the application name is `ODPS_TDDL_TEST_APP` and the table name is `odps_federation_localrun_write`.

Example:

```
drop table if exists text_test_jdbc_external;
CREATE EXTERNAL TABLE if not exists text_test_jdbc_external
(
  colmapping tinyint, --c_tinyint tinyint,
  c_smallint smallint,
  c_int int,
  c_bigint bigint,
  c_utinyint tinyint,
  c_usmallint smallint,
  c_uint int,
  c_ubigint bigint,
  c_boolean tinyint,
  --c_float float, -- in tddl, not recommend float and double type as it may lost precision
  --c_double double,
  c_string string,
  c_datetime datetime,
  c_decimal decimal
)
STORED BY 'com.aliyun.odps.jdbc.JdbcStorageHandler'
location 'jdbc:mysql://dummy_host?table=odps_federation_localrun_write'
TBLPROPERTIES(
'odps.federation.jdbc.insert.type'='simpleInsert',
'odps.federation.jdbc.condition'='c_boolean = 1 and c_int is not null and c_utinyint=127',
'odps.federation.jdbc.colmapping'='colmapping:c_tinyint',
'odps.federation.jdbc.tddl.appname'='ODPS_TDDL_TEST_APP',
'odps.federation.jdbc.tddl.app.access.key'='your tddl app access key',
'odps.federation.jdbc.tddl.app.secret.key'='your tddl app secret key');
```

1.17.2.6.4. Read data from an external table

This topic describes how to read data from an external table and provides an example for reference.

Before you perform complex operations such as GROUP JOIN, we recommend that you import data from external tables to MaxCompute tables. This improves the efficiency of data computations. The following example shows how to import data from an associated MySQL external table to MaxCompute.

Create a MaxCompute table

Example:

```
CREATE TABLE if not exists text_test_jdbc_max_compute
(
  c_tinyint tinyint,
  c_smallint smallint,
  c_int int,
  c_bigint bigint,
  c_utinyint tinyint,
  c_usmallint smallint,
  c_uint int,
  c_ubigint bigint,
  c_boolean tinyint,
  --c_float float,
  --c_double double,
  c_string string,
  c_datetime datetime,
  c_decimal decimal
);
```

Import data to a MaxCompute table

Example:

```
insert OVERWRITE TABLE text_test_jdbc_odps select * from text_test_jdbc_read_external;
```

Relationship between creating an external table and importing data to a MaxCompute table

When you create an external table, a data channel is established between MaxCompute and a MySQL database. MaxCompute does not store data in the MySQL database. If data of the external table is missing from the MySQL database, the data is unavailable in MaxCompute.

If data of the MySQL database is imported to a MaxCompute table, the data is stored in MaxCompute. If data is missing from the MySQL database, you can retrieve the data from the MaxCompute table to which the data is imported.

1.17.2.6.5. Write data to an external table

This topic describes how to write data to an external table.

The column names and data types of the external table must be consistent with those of the database. This ensures that data is correctly written to the external table. For more information about data check actions when loss of precision occurs during data type conversions, see [Syntax](#).

Run the following command to write data to an external table:

```
insert into table text_test_jdbc_external select * from text_test_jdbc_max_compute;
```

 **Note** For MySQL external tables, `Insert INTO MySQL-External-Table` uses the same syntax as `Insert OVERWRITE MySQL-External-Table`. Both statements are executed to append data to external tables. You can use `odps.federation.jdbc.insert.type` to specify the data insertion type. For more information, see [Syntax](#).

1.17.3. External data sources

1.17.3.1. HDFS data source (open-source)

1.17.3.1.1. Overview

This topic describes how to access and process data of open source Hadoop Distributed File System (HDFS) data sources in MaxCompute.

HDFS is the most widely used storage service in the open source community. Most customers use HDFS at the underlying layer of their self-managed big data systems.

MaxCompute uses external tables to access open source HDFS data to facilitate data migration and interact with self-managed customer systems. This reduces customer efforts and costs.

HDFS supports multiple file formats, such as text file, sequence file, RC file, Parquet, and AVRO. Some examples are provided to demonstrate how MaxCompute accesses and processes data of open source HDFS data sources. The text file format is used in the examples.

1.17.3.1.2. Write data to HDFS

1.17.3.1.2.1. Create an external table

This topic describes how to create an external table for an open source Hadoop Distributed File System (HDFS) data source. It also provides an example for reference.

After you compile the `textfiletest` script, execute the following statements to create an external table:

```
set odps.sql.hive.compatible=true;
drop table if exists textfiletest;
CREATE EXTERNAL TABLE if not exists textfiletest
(
  c_int int ,
  c_tinyint tinyint ,
  c_boolean boolean ,
  c_smallint smallint ,
  c_bigint bigint ,
  c_double double ,
  c_float float ,
  --c_time datetime ,
  c_date date ,
  c_timestamp datetime ,
  c_string string
)
STORED as TEXTFILE
location 'hdfs://host:port/user/wbyy/';
-- The file directory /user/wbyy/ is for reference only. Replace it with the actual directory.
```

1.17.3.1.2.2. Write and query data

This topic describes how to use external tables to write and query data from the open source Hadoop Distributed File System (HDFS).

After an external table is created, you can use it the same way that you use a standard table. You can use the `INSERT OVERWRITE`, `INSERT INTO`, and `SELECT` statements to write data and check whether the write operation succeeds. For more information about how to use these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.3.1.3. Read data from HDFS

This topic describes how to read data from open source Hadoop Distributed File System (HDFS) and provides an example for reference.

After you compile the `testfile_read` script, execute the following statements to read data from open source HDFS:

```
set odps.sql.hive.compatible=true;
drop table if exists testfile_read;
CREATE EXTERNAL TABLE if not exists testfile_read
(
  c_int int ,
  c_tinyint tinyint ,
  c_boolean boolean ,
  c_smallint smallint ,
  c_bigint bigint ,
  c_double double ,
  c_float float ,
  --c_time datetime ,
  c_date date ,
  c_timestamp datetime ,
  c_string string
)
STORED as TEXTFILE
location 'hdfs://host:port/user/wbyy/';
-- The file directory /user/wbyy/ is for reference only. Replace it with the actual file directory.
-- Create an external table.
select * from testfile_read;
-- Query and read data from the external table.
```

1.17.3.2. MongoDB data source

1.17.3.2.1. Overview

This topic describes how to access and process data of ApsaraDB for MongoDB data sources in MaxCompute.

ApsaraDB for MongoDB is a stable, reliable, and auto-scaling database service that is fully compatible with MongoDB protocols. ApsaraDB for MongoDB offers a full range of database solutions, such as disaster recovery, backup, restoration, monitoring, and alerting.

MaxCompute can interact with ApsaraDB for MongoDB for joint computations after you create external tables.

Some examples are provided to demonstrate how MaxCompute accesses and processes data of ApsaraDB for MongoDB data sources.

1.17.3.2.2. Preparations

This topic describes the preparations that are required to create an external table and process data of an ApsaraDB for MongoDB data source.

Before you process data of an ApsaraDB for MongoDB data source, you must deploy ApsaraDB for MongoDB. Example:

1. Run the following command to activate ApsaraDB for MongoDB:

```
bin/mongod --dbpath=./db
```

2. Run the following command to go to the ApsaraDB for MongoDB client:

```
bin/mongo --host=${host}
```

3. Run the following command to create a database:

```
use mongodb
```

4. Run the following command to create a username and password:

```
db.createUser({user: '${user}', pwd: '${password}', roles: [{role:'readWrite',db:'mongodb'}]})
```

5. Run the following command to check whether the user is created. If 1 is returned, the user is created.

```
db.auth('${user}', '${password}')
```

1.17.3.2.3. Write data to MongoDB

1.17.3.2.3.1. Create an external table

This topic describes how to create external tables for ApsaraDB for MongoDB data sources and provides an example for reference.

Run the following command to create a collection in ApsaraDB for MongoDB:

```
db.createCollection("${tablename}", { capped : true, autoIndexId : true, size : 6142800, max : 10000 } )  
-- The values of the size and max parameters are for reference only. Replace them with the actual values.
```

After the collection is created, execute the following statements to create an external table:

```
set odps.sql.hive.compatible=true;  
drop table if exists mongo_table_external;  
CREATE external TABLE if not exists mongo_table_external  
(  
  id string,  
  name string  
)  
STORED BY 'com.mongodb.hadoop.hive.MongoStorageHandler'  
location "mcfed:mongodb://${user}:${password}@host:port/mongodb.${tablename}"  
TBLPROPERTIES(  
  "mcfed.mongo.input.split_size"="2",  
  -- The value of input.split_size is for reference only. Replace it with the actual value.  
  "mcfed.location"="mongodb://${user}:${password}@host:port/mongodb.${tablename}",  
  "mcfed.mongo.input.uri"="mongodb://${user}:${password}@host:port/mongodb.${tablename}",  
  "mcfed.mongo.output.uri"="mongodb://${user}:${password}@host:port/mongodb.${tablename}"  
);
```

1.17.3.2.3.2. Write and query data

This topic describes how to use external tables to write and query ApsaraDB for MongoDB data.

After an external table is created, you can use it the same way you use a standard table. You can execute the INSERT OVERWRITE, INSERT INTO, and SELECT statements to write data and check whether the write operation succeeded. For more information about the usage of these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.3.2.4. Read data from ApsaraDB for MongoDB

This topic describes how to read data from ApsaraDB for MongoDB and provides an example for reference.

After a row of data is inserted into an existing dataset, run the following command to read data from ApsaraDB for MongoDB:

```
set odps.sql.hive.compatible=true;
drop table if exists mongo_read_external;
CREATE external TABLE if not mongo_read_external
(
  id string,
  name string
)
STORED BY 'com.mongodb.hadoop.hive.MongoStorageHandler'
location "mcfed:mongodb://${user}:${password}@host:port/mongodb.${tablename}"
TBLPROPERTIES(
  "mcfed.mongo.input.split_size"="2",
  -- The value of input.split_size is for reference only. Replace it with the actual value.
  "mcfed.location"="mongodb://${user}:${password}@host:port/mongodb.${tablename}",
  "mcfed.mongo.input.uri"="mongodb://${user}:${password}@host:port/mongodb.${tablename}"
);
-- Create an external table.
select * from mongo_external;
-- Query and read data from the external table.
```

1.17.3.3. HBase data source

1.17.3.3.1. Overview

This topic describes how to access and process data of ApsaraDB for HBase data sources in MaxCompute.

ApsaraDB for HBase is a distributed database based on Hadoop. It can store petabytes of big data and be used in scenarios that require high-throughput random read and write operations.

MaxCompute allows you to create external tables to interact with ApsaraDB for HBase for joint computations.

The following topics provide examples to show how MaxCompute accesses and processes data of ApsaraDB for HBase data sources.

1.17.3.3.2. Write data to HBase

1.17.3.3.2.1. Write data to ApsaraDB for HBase by using an online API

Create an external table

This topic describes how to create an external table of ApsaraDB for HBase data sources. It also provides an example for reference.

Run the following command to create an external table:

```
set odps.sql.hive.compatible=true;
CREATE EXTERNAL TABLE if not exists hbase_table_external
(
  id string,
  cfa string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('mcfed.hbase.table.name'='${table.name}', 'mcfed.hbase.columns.mapping'=':key,cf:a')
-- cf is the column family in the external table of ApsaraDB for HBase.
location 'hbase://host:port'
TBLPROPERTIES('hbase.table.name'='${table.name}', 'hbase.columns.mapping'=':key,cf:a', 'mcfed.zookeeper.session.timeout'='30', 'mcfed.hbase.client.retries.number'='1', 'mcfed.hbase.zookeeper.quorum'='${host}', 'mcfed.hbase.zookeeper.property.clientPort'='${port}');
-- The values of the zookeeper.session.timeout and hbase.client.retries.number parameters are for reference only.
Replace them with actual values.
```

Write and query data

This topic describes how to use external tables to write and query ApsaraDB for HBase data.

After an external table is created, you can use it the same way that you use a standard table. You can use the INSERT OVERWRITE, INSERT INTO, and SELECT statements to write data and check whether the write operation succeeds. For more information about the usage of these statements, see [Update the data of a table](#) and [SELECT](#).

1.17.3.3.2.2. Write data to ApsaraDB for HBase by using bulk load

This topic describes how to write data to ApsaraDB for HBase by using the bulk load method. It also provides an example for reference.

You can use bulk load to import large amounts of data to ApsaraDB for HBase. In the bulk load process, HFiles of ApsaraDB for HBase are generated and then moved to the directory of ApsaraDB for HBase. Compared with the method of calling the put method of ApsaraDB for HBase to write data, the bulk load method provides higher processing efficiency and has less impact on the running of ApsaraDB for HBase. This is because this method bypasses region servers and write-ahead logging (WAL).

MaxCompute performs the following steps to write data to an external table of ApsaraDB for HBase: 1. Sort data in the source table based on row keys. 2. Write data in the HFile format to Hadoop Distributed File System (HDFS). 3. Move HFiles to the directory of ApsaraDB for HBase and notify region servers to load the data.

Create an external table

Run the following command to create an external table:

```

set odps.sql.hive.compatible=true;
CREATE EXTERNAL TABLE if not exists test_hfile_hbase_external
(
  id string,
  cfa string,
  cfb string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('mcfed.hbase.table.name'='${table.name}','mcfed.hbase.columns.mapping'=':key,cf:a,cf:b')
-- cf is the column family in the external table of ApsaraDB for HBase.
location 'hbase://host:port'
TBLPROPERTIES('hbase.table.name'='${table.name}','hbase.columns.mapping'=':key,cf:a,cf:b', 'hfile.temp.path'='hdfs://hbase-cluster/hfiletmp', 'hadoop.user.name'='admin', 'mcfed.fs.defaultFS'='hdfs://hbase-cluster', 'mcfed.dfs.nameservices'='hbase-cluster', 'mcfed.dfs.ha.namenodes.hbase-cluster'='nn1,nn2', 'mcfed.dfs.namenode.rpc-address.hbase-cluster.nn1'='j63g09343.sqa.eu95:8020', 'mcfed.dfs.namenode.rpc-address.hbase-cluster.nn2'='j63g09355.sqa.eu95:8020', 'mcfed.dfs.client.failover.proxy.provider.hbase-cluster'='org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider', 'mcfed.zookeeper.session.timeout'='30', 'mcfed.hbase.client.retries.number'='1', "mcfed.hbase.zookeeper.quorum"='${host}', "mcfed.hbase.zookeeper.property.clientPort"='${port}');
-- The values of the zookeeper.session.timeout and hbase.client.retries.number parameters are for reference only.
Replace them with actual values.

```

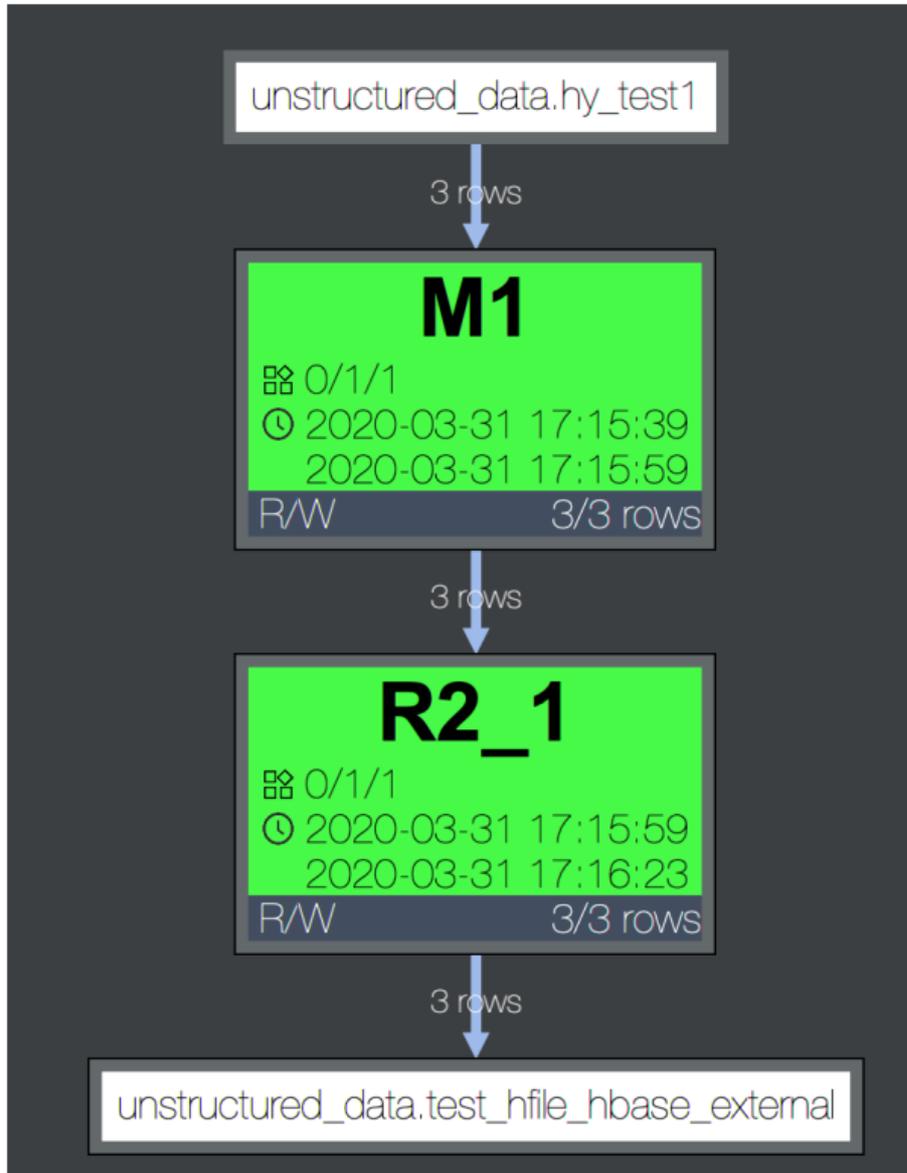
Sort data in the source table based on row keys

The range clustering mechanism of MaxCompute is used to sort data in the source table. If the optimizer generates an execution plan and finds that the table is an external table of ApsaraDB for HBase, the optimizer considers the table as a range-clustered table and sorts all data in the source table.

Example:

```
insert overwrite table test_hfile_hbase_external select uuid(), a_name from hy_test1;
```

The following figure shows the execution plan. Data in the source table `hy_test1` is sorted based on row keys. This step is M1 in the execution plan, as shown in the following figure.



Write data in the HFile format to HDFS

The INSERT statement is executed to automatically write data in the HFile format to HDFS. Each Fuxi instance corresponds to a region for the external table of ApsaraDB for HBase. This step is R2_1 in the execution plan, as shown in the preceding figure. In this step, the sorted data is written in the HFile format to HDFS. You must configure a temporary directory of HDFS to store HFiles to which data is written.

Limits on writing data to HFiles:

- The first column must be a row key, which is `:key` in `hbase.columns.mapping`.
- The other columns must belong to the same column family. Each instance has only one HFile.

Move HFiles to the directory of ApsaraDB for HBase and notify region servers to load the data.

After data in the HFiles of all regions for the external table of ApsaraDB for HBase is written to HDFS, you can perform bulk load. In this step, MaxCompute automatically moves HFiles to the directory of ApsaraDB for HBase and notifies region servers to load the data.

You can run the following command to check the data import result:

```
select * from test_hfile_hbase_external;
```

1.17.3.3.3. Read data from ApsaraDB for HBase

This topic describes how to read data from ApsaraDB for HBase and provides an example for reference.

After you create a table on an ApsaraDB for HBase client and insert data into the table, run the following commands to read data from the ApsaraDB for HBase table:

```
set odps.sql.hive.compatible=true;
drop table if exists hbase_read_external;
CREATE EXTERNAL TABLE if not exists hbase_read_external
(
  id string,
  name string,
  a string
)
STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ('mcfed.hbase.table.name'='${table.name}','mcfed.hbase.columns.mapping'=':key,f1:name,f1:a')
-- f1 is the column family in the ApsaraDB for HBase table.
location 'hbase://host:port'
TBLPROPERTIES('hbase.table.name'='${table.name}','hbase.columns.mapping'=':key,f1:name,f1:a','mcfed.zookeeper.session.timeout'='30','mcfed.hbase.client.retries.number'='1','mcfed.hbase.zookeeper.quorum'='${host}','mcfed.hbase.zookeeper.property.clientPort'='${port}');
-- The values of the zookeeper.session.timeout and hbase.client.retries.number parameters are for reference only.
Replace them with actual values.
-- Create an external table.
select * from hbase_read_external;
select count(*) from hbase_read_external;
select a.id,a.name from hbase_read_external a join hbase_test b on a.id=b.id;
-- Query and read data from the external table.
```

1.18. Unstructured data access and processing (inside MaxCompute)

1.18.1. Overview

This topic describes how to access and process unstructured volume data by using external tables.

If MaxCompute stores unstructured data as volumes, the data cannot be processed in MaxCompute. You must export the data to an external system for processing.

To address this issue, MaxCompute uses external tables to access various data types. MaxCompute uses external tables to read and write data volumes and processes unstructured data from external data sources, such as Object Storage Service (OSS).

The following topics describe how to create and access volume external tables.

1.18.2. Create a volume external table

1.18.2.1. Syntax

This topic describes the syntax that is used to create a volume external table.

You can first execute the CREATE EXTERNAL TABLE statement to create an external table. The following example shows the syntax:

```
DROP TABLE [IF EXISTS] <external_table_name>;
Create External Table [IF NOT EXISTS] <external_table_name>
(<column schemas>)
[PARTITIONED BY (partition column schemas)]
STORED BY '<StorageHandler>'
[WITH SERDEPROPERTIES (
    'name'='value'
)]
LOCATION 'volume://...'
[USING '<ResourceName>']
;
```

Parameters:

- **STORED BY:** Two built-in storage handlers `com.aliyun.odps.CsvStorageHandler` and `com.aliyun.odps.TsvStorageHandler` are supported. They can be used to read and write CSV files where the column delimiter is a comma and the row delimiter is `\n` or TSV files where the column delimiter is `\t` and the row delimiter is `\n`. If the built-in storage handlers cannot be used, you can build a custom storage handler.
- **WITH SERDEPROPERTIES:** specifies table attributes such as delimiters for a custom storage handler.
- **LOCATION:** the location format of the table.

Format:

```
volume://[project_name]/volume_name/partition_value
```

Example:

```
volume://test_project/volume_data/20190102
```

`project_name` is optional. If `project_name` is not specified, the current project is used to obtain volume data after the DML SQL statement is executed. In the preceding example, if the current project is `myproject`, you can use the following location format:

```
volume:///volume_data/20190102
```

Notice

- The location of a non-partitioned table must point to a volume partition, instead of the volume.
- The location of a partitioned table must point to the volume.
- The volume path cannot contain an equal sign (=) and does not support the default standard partition path `ds=2017071` that is used when a partition is created. The partition path must be customized. The custom partition path can be any path supported by the volume. For example, if the partition path is `20190102`, the path combined with volume path can be `volume://test_project/volume_data/20190102`.

- **USING:** specifies a storage handler. To use a custom storage handler, you must export the custom storage

handler as a JAR package and then add it to MaxCompute as a JAR resource.

1.18.2.2. Use a built-in storage handler to create a table

This topic describes how to use a built-in storage handler to create a table.

You can use a built-in storage handler to create a partitioned table or non-partitioned table.

Create a non-partitioned table

Example:

```
DROP TABLE IF EXISTS volume_ext;
Create External Table volume_ext
(
  key string,
  value string
)
STORED BY 'com.aliyun.odps.CsvStorageHandler'--This is the built-in storage handler.
LOCATION 'volume://test_project/volume_data/20190102'
;
```

Create a partitioned table

Example:

```
DROP TABLE IF EXISTS volume_ext_pt;
Create External Table volume_ext_pt
(
  key string,
  value string
)
PARTITIONED BY (ds string)
STORED BY 'com.aliyun.odps.CsvStorageHandler'--This is the built-in storage handler.
LOCATION 'volume://test_project/volume_data'
;
ALTER TABLE volume_ext_pt DROP IF EXISTS PARTITION (ds="20190102");
ALTER TABLE volume_ext_pt ADD PARTITION (ds="20190102") LOCATION "volume://test_project/volume_data/20190102";
```

1.18.2.3. Use a custom StorageHandler to create a table

This topic describes how to use a custom StorageHandler to create a table.

If the built-in StorageHandlers are unable to meet your business requirements, you can customize a StorageHandler by using Java and specify some attributes of the Volume external table through which you want to process data.

The following example shows how to create a table:

Assume that the data type of the table that you want to create is TXT and the column delimiter is |. You can perform the following steps to create an external table:

1. Use the MaxCompute Studio or MaxCompute Eclipse development tool to customize Java classes.
2. Export the JAR package. In this example, the package name is odps-volume-example.jar.
3. Run the following command to add the JAR package to MaxCompute as a resource:

```
add jar odps-volume-example.jar -f;
```

4. Run the following commands to create an external table:

```
DROP TABLE IF EXISTS volume_ext;
Create External Table volume_ext
(
  key string,
  value string
)
STORED BY 'com.aliyun.odps.udf.example.text.TextStorageHandler'
WITH SERDEPROPERTIES (
  'delimiter'='|'
)
LOCATION 'volume://myproject/volume_data/20190102'
USING 'odps-volume-example.jar'
;
```

 **Note** After the external table is created, you can process the volume data by using the external table.

1.18.3. Access a Volume external table

This topic describes how to access a Volume external table.

Volume external tables can be accessed in the same way that you access a MaxCompute table. Example:

```
select key,value from volume_ext_pt where ds="20190102";
```

1.19. Multi-region cluster deployment on MaxCompute

1.19.1. Overview

MaxCompute supports multi-region deployment. This topic describes how to deploy control clusters and compute clusters.

Centralized deployment of control clusters is designed for implementing resource configuration and computing task management.

Independent deployment of compute clusters for each region is designed for implementing project creation and compute task issuance.

1.19.2. Features of multi-region deployment

This topic describes the features of multi-region deployment on MaxCompute.

Multi-region deployment on MaxCompute has the following features:

- One MaxCompute service instance can manage multiple clusters that are deployed in different regions.
- Data exchanges between clusters are completed within MaxCompute, and data replication and synchronization between clusters are managed based on configured policies.
- Metadata is stored in a unified manner, which leads to high requirements for infrastructure such as network

- connections between data centers.
- A unified account system is required.
- The development systems for big data applications, such as DataWorks, are used for all clusters in all regions.
- MaxCompute must run in multi-cluster mode to support multi-region deployment.

 **Note** The conditions and limits on changing the cluster mode must meet the following requirements:

- The network bandwidth must be sufficient to support multi-region data synchronization and link redundancy.
 - The central-region control cluster has a relatively high latency requirement for basic services such as DNS and Tablestore. We recommend that you deploy basic services in the same data center to keep the network latency within 5 ms.
 - The network latency between control clusters in the central region and compute clusters in other regions is within 20 ms.
 - Clocks must be synchronized between clusters in different regions and between machines in the same cluster.
 - The network bandwidth must be sufficient to support data replication among clusters.
 - Alibaba Cloud DNS is required.
 - Machines in different clusters can communicate with each other, and the clusters have similar network infrastructure (1-Gigabit or 10-Gigabit).
- The O&M and upgrades for multi-region deployment are different from those for single-cluster deployment. Multi-region deployment requires higher on-site O&M capabilities.

1.19.3. Instructions on multi-region deployment

This topic describes the multi-region deployment instructions for MaxCompute.

Instructions on multi-region deployment:

- Computing tasks on MaxCompute are implemented in a cluster based on data distribution within MaxCompute. Cross-cluster replication and direct reads are performed based on the cluster configuration and distribution of data.
- Data replication and synchronization can be performed between clusters by table or partition based on the actual cluster configuration.
- You must plan the relationship between MaxCompute projects and clusters based on your business requirements. You can select one or more clusters for a single project.

 **Note** If you select multiple clusters for a project, data replication is performed among these clusters.

- If cross-cluster computing cannot be implemented for a business in a project, you can distribute the data of this business to different clusters and divide the computing tasks based on data distribution.

 **Note** Cross-cluster data replication and management requires management policies and maintenance.

- The bandwidth between clusters in different remote data centers must be sufficient for data replication or read-only operations.
- The two data centers that are involved in cross-region operations share the bandwidth. As a result, cross-

cluster replication and read-only operations that involve large amounts of data may consume all of the bandwidth between the two data centers, which affects data transmission.

- A cluster failure in the primary data center affects the entire service because the primary data center stores information including metadata and accounts.
- When the secondary data center fails, projects that have data stored on a cluster of this data center and the O&M of services are affected.

Note If project data is independently stored in the primary data center, the failure of the secondary data center does not affect the computing tasks of the project.

- In general scenarios, data is distributed and replicated based on business characteristics, and computing tasks are performed based on data types in different clusters. The combined results of different MaxCompute tasks are applied.

1.19.4. Multi-region deployment examples

1.19.4.1. Synchronize table data among multiple clusters

This topic provides an example of table data synchronization between multiple clusters in a multi-region deployment scenario.

Prerequisites

- When you create a project in the AdminConsole, you have selected two clusters and set one as the default cluster. The following figure shows an example.

Note

- In this example, the project is multiregion, and AT-MDU-TEST and AT-5KN are selected for it, with AT-MDU-TEST as the default cluster.
- The URL of Apsara Stack MaxCompute AdminConsole is `http://{odps_ag}:9090`, which is port 9090 of MaxCompute AG.
- The operation entry to create the project is **MaxCompute Configuration > Project Management > Create Project**.

- You have configured cross-cluster replication for the project. The following example shows how to configure it.
 - i. Choose **MaxCompute Configuration > Global Cross-cluster Replication Configuration** to go to the **Global Cross-cluster Replication Configuration** page.
 - ii. In the Add Item part of the **InfoBetweenClusters** section, set key to **AT-5KN#AT-MDU-TEST**, set both **availableBandwidth** and **totalBandwidth** to **2000**, and click **add**.
 - iii. Choose **MaxCompute Configuration > Project Management**. Find multiregion and click **Cross-cluster Replication Configuration**.
 - iv. In the dialog box that appears, configure the parameters, as shown in the following figure.
 - v. Click **Save** and then click **Start Replication**.

Procedure

Example:

1. In the AG of the default cluster AT-MDU-TEST, construct an upload tunnel data file in the directory of the same level as console, such as `echo "testtest" > uploaddata`.

- Go to the console command line and run the following commands to create a table and insert data into the table:

```
use multiregion;
create table t1 (s string);
tunnel upload uploaddata t1;
```

- Run the following command to check whether the data is inserted into the table:

```
select * from t1;
```

- In the AG of AT-MDU-TEST, view the Apsara Distributed File System directory for this table. Run the following command to check whether data exists in the directory:

```
pu ls product/aliyun/odps/multiregion/data/t1
```

- In the AG of AT-5KN, view the Apsara Distributed File System directory for this table. Run the following command to check whether data exists in the directory:

```
pu ls /apsara/odps/mdutesting/multiregion/data/t1
```

- If data is displayed after you perform Steps 4 and 5, cross-cluster data synchronization is successful.

1.19.4.2. Query the status of data synchronization between primary and secondary clusters

This topic provides an example on how to query the status of data synchronization between the primary and secondary clusters in multi-region deployment scenarios.

Prerequisites

- When you create a project in the AdminConsole, you have selected two clusters and set one as the default cluster. The following figure shows an example.

Note

- In this example, the project is multiregion, and AT-MDU-TEST and AT-5KN are selected for it, with AT-MDU-TEST as the default cluster.
- The URL of Apsara Stack MaxCompute AdminConsole is `http://{odps_ag}:9090`, which is port 9090 of MaxCompute AG.
- The operation entry to create the project is **MaxCompute Configuration > Project Management > Create Project**.

- You have configured cross-cluster replication for the project. The following example shows how to configure it.
 - Choose **MaxCompute Configuration > Global Cross-cluster Replication Configuration** to go to the **Global Cross-cluster Replication Configuration** page.
 - In the Add Item part of the **InfoBetweenClusters** section, set key to **AT-5KN#AT-MDU-TEST**, set both **availableBandwidth** and **totalBandwidth** to **2000**, and click **add**.
 - Choose **MaxCompute Configuration > Project Management**. Find multiregion and click **Cross-cluster Replication Configuration**.
 - In the dialog box that appears, configure the parameters, as shown in the following figure.
 - Click **Save** and then click **Start Replication**.

Procedure

Example:

1. Compile the Bodychecksync configuration file, which is used to send Admin task.

```
<Instance>
  <Job>
    <Comment>
    </Comment>
    <Priority>1</Priority>
    <Tasks>
      <Admin>
        <Name>task_1</Name>
        <Comment>test</Comment>
        <Config>
          <Property>
            <Name>PROJECT</Name>
            <Value>multiregion</Value>
          </Property>
          <Property>
            <Name>CLUSTER</Name>
            <Value>AT-MDU-TEST</Value>
          </Property>
        </Config>
        <Command>GET_UNREPLICATED_OBJECTS</Command>
      </Admin>
    </Tasks>
    <DAG>
      <Comment/>
      <RunMode>Sequence</RunMode>
    </DAG>
  </Job>
</Instance>
```

2. Compile the Header.

```
content-type: application/xml
```

3. Run the following command to send Admin task:

```
CLTrelease/bin/odpscmd -e "http post /projects/admin_task_project/instances -header=header -content=bodychecksync;"
```

4. Run the following command to check the instance execution results:

```
CLTrelease/bin/odpscmd --project=admin_task_project -e "wait 20180711050550317gnege3ms2;"
```

5. Access the LogView URL generated in Step 4 in a browser.
6. On the page that appears, click **Detail**. The status of data synchronization between the primary and secondary clusters is displayed.

1.19.4.3. Cross-region direct read

This topic provides examples of cross-region direct read operations in multi-region deployment scenarios.

Cross-region direct read can be implemented with or without cross-cluster access.

 **Note** When cross-cluster access is not configured and the data volume is large, cross-region direct read takes a long period of time.

Direct read when cross-cluster access is not configured

Example:

1. In the AdminConsole, create two projects: testmdu and testcross5kn.

 **Note**

- In this example, AT-MDU-TEST and AT-5KN are selected for testmdu, with AT-MDU-TEST as the default cluster. AT-5KN is selected for testcross5kn and serves as the default cluster.
- The URL of Apsara Stack MaxCompute AdminConsole is `http://{odps_ag}:9090`, which is port 9090 of MaxCompute AG.
- The operation entry to create the project is **MaxCompute Configuration > Project Management > Create Project**.

2. Create tables for testmdu and testcross5kn.

```
create table testrep(s string);
create table tablecross5kn (s string);
```

3. Construct some data for tables testrep and tablecross5kn.
4. Run the following commands to read the table data in testmdu directly from testcross5kn:

```
use testcross5kn;
select * from testmdu.testrep;
```

 **Note** If the required data is returned, cross-region direct read is successful.

Direct read when cross-cluster access is configured

Example:

1. In the AdminConsole, create two projects: testmdu and testcross5kn.

 **Note**

- In this example, AT-MDU-TEST and AT-5KN are selected for testmdu, with AT-MDU-TEST as the default cluster. AT-5KN is selected for testcross5kn and serves as the default cluster.
- The URL of Apsara Stack MaxCompute AdminConsole is `http://{odps_ag}:9090`, which is port 9090 of MaxCompute AG.
- The operation entry to create the project is **MaxCompute Configuration > Project Management > Create Project**.

2. Create tables for testmdu and testcross5kn.

```
create table testrep(s string);
create table tablecross5kn (s string);
```

3. Construct some data for tables testrep and tablecross5kn.
4. Configure cross-cluster replication for testmdu.

 **Note** Choose **MaxCompute Configuration > Project Management**. Find testmdu and click **Cross-cluster Replication Configuration**.

5. Run the following command to read the table data in testmdu directly from testcross5kn:

```
use testcross5kn;  
select * from testmdu.testrep;
```

 **Note** If the required data is returned, cross-region direct read is successful.

1.19.4.4. Cross-region JOIN

This topic provides an example of cross-region JOIN in multi-region deployment scenarios.

Example:

1. In Apsara Stack MaxCompute AdminConsole, create two projects: crossregion and crossregion02.

 **Note**

- In this example, select the AT-MDU-TEST and AT-70N compute clusters for the crossregion project. The default cluster for the crossregion project is AT-MDU-TEST. Select only the default cluster AT-70N for the crossregion02 project.
- The endpoint of Apsara Stack MaxCompute AdminConsole is `http://{odps_ag}:9090`. 9090 indicates the port number of ODPS AG.
- To perform this step, choose **MaxCompute Configuration > Project Management > Create Project**.

2. Configure cross-cluster replication for the crossregion project.

 **Note** Choose **MaxCompute Configuration > Project Management**. Find the crossregion project and click **Cross-cluster Replication Configuration**.

3. Create the business table for the crossregion project.

```
create table business(bid bigint,name string,phone string,address string,region string);
```

4. Generate data for the business table.
5. Run the following command to import data into the business table:

```
tunnel upload business business;
```

6. Create the product table for the crossregion02 project.

```
create table if not exists product(pid bigint,name string,type string,color string,bid bigint);
```

7. Generate data for the product table.
8. Run the following command to import data into the product table:

```
tunnel upload product product;
```

9. Run the following command in Apsara Stack MaxCompute AdminConsole to obtain specified data from the business and product tables:

```
select pro.pid,pro.name,pro.type,pro.color,bus.name,bus.phone from crossregion02.product pro join crossregion.business bus on pro.bid=bus.bid where bus.region='Shanghai';
```

 **Note** If the specified data is returned, the cross-region JOIN operation is complete.

1.20. Security solution

1.20.1. Intended users

This section describes the users for which the MaxCompute security solution is intended.

This MaxCompute security solution is intended for all owners and administrators of MaxCompute projects, and users who are interested in the MaxCompute multi-tenant data security system. The MaxCompute multi-tenant data security system supports the following features:

- User authentication
- User and authorization management of projects
- Cross-project resource sharing
- Project data protection

1.20.2. Quick start

This topic describes the procedure of the MaxCompute security solution. This provides guidance for you to perform MaxCompute security operations.

Add a user and grant permissions to the user

Scenario: Jack is the administrator of the prj1 project. Alice, who has an Alibaba Cloud account (alice@aliyun.com), joins the prj1 project. Alice applies for the permissions to create and view tables and submit jobs.

Statements:

```
use prj1
add user aliyun$alice@aliyun.com
-- Add a user.
grant List, CreateTable, CreateInstance on project prj1 to user aliyun$alice@aliyun.com
-- Grant permissions to the user.
```

 **Note** Only the project administrator can perform this operation.

Add a role and grant permissions to the role by using ACL-based authorization

Scenario: Jack is the administrator of the prj1 project. Three new members Alice, Bob, and Charlie join the project as data reviewers. The members apply for the permissions to view tables, submit jobs, and read the user profile table.

The project administrator can use ACL-based authorization for objects in this scenario. Statements:

```
use prj1
add user aliyun$alice@aliyun.com
-- Add users.
add user aliyun$bob@aliyun.com
add user aliyun$charlie@aliyun.com
create role tableviewer
-- Create the tableviewer role.
grant List, CreateInstance on project prj1 to role tableviewer
-- Grant permissions to the tableviewer role.
grant Describe, Select on table userprofile to role tableviewer
grant tableviewer to aliyun$alice@aliyun.com
-- Assign the tableviewer role to users.
grant tableviewer to aliyun$bob@aliyun.com
grant tableviewer to aliyun$charlie@aliyun.com
```

Package and share resources

Scenario: Jack is the administrator of the prj1 project. John is the administrator of the prj2 project. Jack wants to share some resources of the prj1 project, such as the datamining.jar file and the sampletable table to the prj2 project owned by John. If Bob in the prj2 project wants to access these resources, John can use ACL- or policy-based authorization to grant permissions to Bob, without the assistance of Jack.

Statements:

1. Jack, the administrator of the prj1 project, creates a resource package in the project.

```
use prj1
create package datamining
-- Create a package.
add resource datamining.jar to package datamining
-- Add resources to the package.
add table sampletable to package datamining
-- Add a table to the package.
allow project prj2 to install package datamining
-- Share the package to the prj2 project.
```

2. John installs the package in the prj2 project.

```
use prj2
install package prj1.datamining
-- Install the package.
describe package prj1.datamining
-- View the resources of the package.
```

3. John grants Bob the permission to use the package.

```
use prj2
grant Read on package prj1.datamining to user aliyun$bob@aliyun.com
-- Use ACL-based authorization to grant Bob the permission to use the package.
```

 **Note** For more information about cross-project resource sharing, see [Cross-project resource sharing](#).

Configure project data protection

Scenario: Jack is the administrator of the prj1 project. This project contains sensitive data, such as user IDs and shopping records. The project also stores a large number of data mining algorithms that are proprietary. Jack wants to protect the sensitive data and algorithms in the project and allows only users in the project to access the data. In this case, the data cannot be transferred to other projects.

Statements:

```
use prj1
set ProjectProtection=true
-- Enable project data protection.
```

If project data protection is enabled, data in the project can be shared only within the project. Data cannot be transferred to other projects. In some cases, for example, Alice wants to export data tables to other projects for business purposes. This operation is approved by the project administrator. MaxCompute provides two methods to export data from a project for which project data protection is enabled.

Method 1: Create an exception policy. For more information, see [Data export methods when project protection is enabled](#).

1. Create a policy file. For example, create the `/tmp/exception_policy.txt` file that only allows Alice to use SQL tasks to export the t1 table from the prj1 project. Content in the `exception_policy.txt` file:

```
{
  "Version": "1",
  "Statement": [{
    "Effect": "Allow",
    "Principal": "ALIYUN$alice@aliyun.com",
    "Action": ["odps:Describe", "odps:Select"],
    "Resource": "acs:odps:*:projects/prj1/tables/t1",
    "Condition": {
      "StringEquals": { "odps:TaskType": "SQL"
    }
  }
}]
```

2. Method 1: Configure an exception policy.

```
use prj1
-- Enable project data protection and configure an exception policy.
set ProjectProtection=true with exception /tmp/exception_policy.txt
```

Note To configure an exception policy, make sure that the principal is not granted the DROP and CREATE permissions or the UPDATE permission on resources. This prevents data leaks caused by time-of-check to time-of-use (TOC2TOU).

Method 2: Configure trusted projects. Configure the prj2 project as a trusted project of the prj1 project to enable data transfer from the prj1 project to the prj2 project. For more information, see [Data export methods when project protection is enabled](#).

```
use prj1
add trustedproject prj2
```

Note The package-based resource sharing and project data protection mechanisms are two orthogonal security mechanisms.

In MaxCompute, resource sharing has a higher priority than project data protection. If an object in a protected project is shared with other projects by using the package mechanism, cross-project access to this object is not subject to the rules for project data protection.

1.20.3. User authentication

This topic describes user authentication for MaxCompute security solutions.

User authentication is used to verify the identity of a request sender. Authentication typically includes:

- Verifying the identity of a message sender
- Checking whether the message was tampered with before it is received.

1.20.4. Project user and authorization management

1.20.4.1. Overview

This topic describes user management, role management, and authorization for projects.

Projects are the foundation of the MaxCompute multi-tenant system and the basic unit of data management and computing. If you create a project, you are the owner of this project. You can manage all objects in the project, such as tables, instances, resources, and user-defined functions (UDFs). Users, except the project owner, can access objects in the project only after these users obtain approvals from the project owner.

For example, if Alice is the owner of the test_project project, and another user from the project team of Alice requests to access the resources in the test_project project, Alice can use the methods described in this topic to perform user management and authorization. If the user is not from the project team of Alice, we recommend that Alice implement cross-project resource sharing. For more information, see [Cross-project resource sharing](#).

1.20.4.2. User management

This topic describes how to manage users in projects.

Add a user

If the project owner, Alice, decides to authorize another user, Alice must add the user to this project first. Alice can authorize only the users in the project.

Run the following command to add a user:

```
add user <full_username>
-- Add a user to a project.
```

Remove a user

If a user leaves the project team, Alice must remove the user from the project. After a user is removed from the project, the user no longer has permissions to access resources in the project.

Run the following command to remove a user from a project:

```
remove user <full_username>
-- Remove a user from a project.
```

Note

- After a user is removed, the user no longer has permissions to access resources in the project.
- Before you remove a user who has been assigned a role, you must first revoke the role. For information about roles, see [Role management](#).
- After a user is removed, the ACL-based authorization related to the user is retained. The policy-based authorization at the role level is revoked, but the policy-based authorization at the project level is retained. If the user is added to the project again, the historical permissions granted by using ACL-based authorization are activated again.
- Removed users may be added to the original projects as different identities. This poses data security risks. If you are a project owner or use an account that has been assigned the Admin or Super_administrator role, you can run the `purge privs from user <username>;` command to permanently delete the permission information of removed users. The permission information includes the permissions granted by using ACL-, label-, and policy-based authorization information. If you do not remove the relevant users from the project, the following error is returned after you run the `purge privs from user <username>;` command: `Principal still exist in the project`

1.20.4.3. Role management

This topic describes the operations that you can perform to manage roles in projects.

A role is a collection of access permissions. A role can be used to assign the same permissions to a group of users. Role-based authorization can greatly simplify the authorization process and reduce authorization management costs. We recommend that you use role-based authorization to authorize users.

An admin role is automatically created when a project is created. This role is granted permissions to access all objects of the project, manage users and roles, and authorize users and roles. Compared with the project owner, the user who is assigned the admin role cannot assign another user with the admin role, configure security rules for a project, or change the authentication model of the project. Permissions of the admin role cannot be modified.

Commands that are used for role management:

```
create role <rolename>
-- Create a role.
drop role <rolename>
-- Delete a role.
grant <rolename> to <username>
-- Assign a role to a user.
revoke <rolename> from <username>
-- Revoke a role from a user.
```

Note When you delete a role, MaxCompute checks whether users who are assigned this role exist. If these users exist, the role fails to be deleted. To delete the role, you must revoke this role from all users.

1.20.4.4. ACL-based authorization actions

This topic describes the actions of ACL-based authorization for projects.

Authorization usually involves three elements: subject, object, and action. In a MaxCompute project, a subject is a user and various types of objects are included. Different types of objects support different actions.

The following table describes the types of objects and actions supported by these objects in a MaxCompute object.

Types and supported actions of objects

Object	Action	Description
Project	Read	Views project information (excluding project objects), such as the creation time.
Project	Write	Updates project information (excluding project objects), such as comments.
Project	List	Views objects of all types in a project.
Project	CreateTable	Creates tables in a project.
Project	CreateInstance	Creates instances in a project.
Project	CreateFunction	Creates functions in a project.
Project	CreateResource	Creates resources in a project.
Project	CreateJob	Creates jobs in a project.
Project	CreateVolume	Creates volumes in a project.
Project	All	Supports all the preceding actions on projects.
Table	Describe	Reads the metadata of a table.
Table	Select	Reads data of a table.
Table	Alter	Modifies the metadata of a table. Adds or deletes partitions.
Table	Update	Overwrites or adds data of a table.
Table	Drop	Deletes a table.
Table	All	Supports all the preceding actions on tables.
Function	Read	Reads data and executes functions.
Function	Write	Updates data.
Function	Delete	Deletes data.
Function	All	Supports all the preceding actions on functions.
Resource, Instance, Job, Volume	Read	Reads data of resources, instances, jobs, and volumes.
Resource, Instance, Job, Volume	Write	Updates resources, instances, jobs, and volumes.

Object	Action	Description
Resource, Instance, Job, Volume	Delete	Deletes resources, instances, jobs, and volumes.
Resource, Instance, Job, Volume	All	Supports all the preceding actions on resources, instances, jobs, and volumes.

Note In the preceding table, the CREATE TABLE permission on projects and the SELECT, ALTER, UPDATE, and DROP permissions on tables must be used with the CREATE INSTANCE permission on projects. If these permissions are not used with the CREATE INSTANCE permission, these permissions may become ineffective.

After you add users or create roles, you must grant permissions to these users or roles. ACL-based authorization of MaxCompute is an object-based authorization. An access control list (ACL) that contains the authorization data is considered a resource of an object. Authorization can be performed only when the object exists. If the object is deleted, the ACL is automatically deleted.

ACL-based authorization of MaxCompute is performed by running the GRANT or REVOKE command defined in SQL-92. These commands are used to grant or revoke permissions to or from objects in a project.

Command syntax:

```
grant actions on object to subject
revoke actions on object from subject
actions ::= action_item1, action_item2, ...
object ::= project project_name | table schema_name | instance inst_name | function func_name | resource res_name
subject ::= user full_username | role role_name
```

Note

The commands used for ACL-based authorization do not support the [WITH GRANT OPTION] clause. If user A authorizes user B to access an object, user B cannot authorize user C to access the same object. Therefore, all authorization actions must be performed by one of the following types of users:

- Project owner
- Users who are assigned the admin role in a project
- Object creators in a project

Example of ACL-based authorization:

Scenario: Users with Alibaba Cloud accounts alice@aliyun.com and bob@aliyun.com are new members of the test_project project. In the test_project project, they must be able to submit jobs, create data tables, and view existing objects of the project. The administrator performs the following authorization actions:

```
use test_project
-- Open a project.
security
add user aliyun$alice@aliyun.com
-- Add alice@aliyun.com to the project.
add user aliyun$bob@aliyun.com
-- Add bob@aliyun.com to the project.
create role worker
-- Create a role.
grant worker TO aliyun$alice@aliyun.com
-- Assign the role to alice@aliyun.com.
grant worker TO aliyun$bob@aliyun.com
-- Assign the role to bob@aliyun.com.
grant CreateInstance, CreateResource, CreateFunction, CreateTable, List ON PROJECT test_project TO ROLE worker
-- Grant permissions to the role.
```

1.20.4.5. View permissions

MaxCompute allows you to view various types of permissions. The permissions include the permissions of users or roles and the access control list (ACL) of specified objects.

View the permissions of a user

Syntax:

```
show grants
-- View the access permissions of a user.
show grants for <username>
-- View the access permissions of a specified user. Only project owners and administrators can perform this operation.
```

View the permissions of a role

Syntax:

```
describe role <rolename>
-- View the access permissions that are granted to a specified role.
```

View the ACL of a specified object

Syntax:

```
show acl for <objectName> [on type <objectType>]
-- View the ACL of a specified object.
```

 **Note** If [on type <objectType>] is not specified, the object type is table.

Show permissions

MaxCompute uses the following characters to indicate the permissions of users or roles:

- A: Allow. Access is allowed.

- D: Deny. Access is denied.
- C: with Condition. This is conditional authorization. This character appears only for policy-based authorization. For more information, see [Condition block structure](#).
- G: with Grant Option. Permissions on objects can be granted.

Example:

```
odps@test_project> show grants for aliyun$odpctest1@aliyun.com
[roles]
dev
Authorization Type: ACL
[role/dev]
A projects/test_project/tables/t1: Select [user/odpctest1@aliyun.com]
A projects/test_project: CreateTable | CreateInstance | CreateFunction | List
A projects/test_project/tables/t1: Describe | Select
Authorization Type: Policy
[role/dev]
AC projects/test_project/tables/test_*: Describe
DC projects/test_project/tables/alifinance_*: Select [user/odpctest1@aliyun.com]
A projects/test_project: Create* | List
AC projects/test_project/tables/alipay_*: Describe | Select
Authorization Type: ObjectCreator
AG projects/test_project/tables/t6: All
AG projects/test_project/tables/t7: All
```

1.20.5. Cross-project resource sharing

1.20.5.1. Overview

This topic describes resource sharing across projects.

You are the owner or administrator (with the admin role assigned) of a project, and a user applies for accessing resources of your project. If the applicant is a member of your project, we recommend that you use the user and authorization management features for your project. For more information, see [Project user and authorization management](#). Otherwise, you can share resources with the user across projects by using packages.

A package is used to share data and resources across projects. You can use a package to implement cross-project user authorization. The following example describes a scenario in which the package mechanism can be used.

Members of the Alifinance project need to access data of the Alipay project. The Alipay project administrator adds members of the Alifinance project to the Alipay project, and then grants the new members common permissions on the Alipay project. For security purposes, the Alipay project administrator does not want to authorize every user of the Alifinance project. A mechanism is required to allow the Alifinance project administrator to control access to the authorized objects.

If the package mechanism is used, the Alipay project administrator can package the objects that the Alifinance team needs to access, and then allow the package to be installed in the Alifinance project. After the package is installed, the Alifinance project administrator can determine whether to grant permissions on the package to members in the Alifinance project.

A package involves two subjects: package creator and package user. The package creator provides resources. The package creator packages the resources to be shared and the permissions to access these resources, and grants the package user the permissions to install and use the package. The package user consumes the resources. After the package user installs the package published by the package creator, the package user can directly access the resources.

The following topics describe the operations that can be performed by a package creator and package user.

1.20.5.2. Package usage

1.20.5.2.1. Operations for package creators

This topic describes the operations that package creators can perform.

Create a package

Run the following command to create a package:

```
create package <pkpname>
```

Delete a package

Run the following command to delete a package:

```
delete package <pkpname>
```

Add a resource that you want to share to a package

Run the following command to add a resource to a package:

```
add project_object to package package_name [with privileges <privileges>]
remove project_object from package package_name
project_object ::= table table_name | instance inst_name | function func_name | resource res_name
privileges ::= action_item1, action_item2, ...
```

Note

- The object type cannot be project. You cannot use a package of a project to create objects in other projects.
- In addition to the objects, the operation permissions on the objects are added to the package. If you do not specify permissions for an object by using [with privileges privileges], the object is read-only. You are granted only the READ, DESCRIBE, and SELECT permissions on the read-only objects. An object and its permissions are considered a whole. You can add or delete resources in a package. The permissions are revoked after resources are added or deleted.

Allow a project to use a package

Run the following command to allow a project to use a package:

```
allow project <prjname> to install package <pkpname> [using label <number>]
```

Revoke the permission for a project to use a package

Run the following command to revoke the permission for a project to use a package:

```
disallow project <prjname> to install package <pkpname>
```

View the packages that are already created and installed

Run the following command to view the packages that are already created and installed:

```
show packages
```

View details of a package

Run the following command to view details of a package:

```
describe package <pkgname>
```

1.20.5.2.2. Operations of package users

This topic describes the operations that are performed by package users.

The installed package is a type of independent object in MaxCompute. Resources in a package are those of other projects that are shared with you. To access resources in a package, you must have the permission to read the package. If you do not have this permission, request the project owner or user admin to grant the permission to you. The project owner or user admin can grant the permission by using ACL- or policy-based authorization.

Example of ACL-based authorization: If you are the project owner or user admin, you can run the following commands to allow a user with the Alibaba Cloud account `odps_test@aliyun.com` to access resources in a package.

```
use prj2 security
install package prj1.testpkg
grant read on package prj1.testpackage to user aliyun$odps_test@aliyun.com
```

Example of policy-based authorization: If you are the project owner or user admin, you can run the following commands to allow all users in the `prj2` project to access resources in the package.

```
use prj2
install package prj1.testpkg
put policy /tmp/policy.txt
```

The `policy.txt` file in the `/tmp` directory contains the following content:

```
{
  "Version": "1", "Statement": [{
    "Effect": "Allow",
    "Principal": "*",
    "Action": "odps:Read", "Resource": "acs:odps:*:projects/prj2/packages/prj1.testpkg"
  }]
}
```

Install a package

Run the following command to install a package:

```
install package <pkgname>;
```

 **Note** In the preceding command, `pkgname` must be in the format of `<projectName>.<packageName>`.

Uninstall a package

Run the following command to uninstall a package:

```
uninstall package <pkgname>;
```

 **Note** In the preceding command, pkgname must be in the format of <projectName>.<packageName>.

View packages

Run the following commands to view packages:

```
show packages
-- View the packages that you have already created and installed.
describe package <pkgname>
-- View details of a package.
```

1.20.6. Project protection

1.20.6.1. Overview

This topic describes the project data protection mechanism.

Some enterprises, such as financial institutions and military enterprises, have high data security requirements. For example, their employees can perform their jobs only in the workplace, and are not allowed to take work materials out of the office. All USB ports on office computers are disabled. These measures aim to prevent leaks of sensitive data.

For example, you are a MaxCompute project administrator in charge of a project with sensitive data. The data must not be shared to other projects. In this case, you are required to perform configurations and operations for project data protection.

1.20.6.2. Project data protection

This topic describes the project data protection mechanism of MaxCompute.

MaxCompute provides a project data protection mechanism. This mechanism can forbid all operations that may cause data transfer to other projects. You need only to run the following command to enable the data protection mechanism for your project.

```
set security.ProjectProtection=true;
-- Set security.ProjectProtection to true. This ensures that data can be used within a project and cannot be transferred to other projects. This setting does not take effect on data in external tables.
-- You can also run the following commands to enable the project data protection mechanism.
set security.ProjectProtection=true;
set project IDENTIFY_EXTERNAL_TABLE_WRITE_AS_DATALEAK=true;
-- After you set security.ProjectProtection to true, set project IDENTIFY_EXTERNAL_TABLE_WRITE_AS_DATALEAK to true. This means that project data cannot be written to external storage sources by using external tables.
```

After the project data protection mechanism is enabled, data of your project cannot be transferred to other projects.

By default, `security.ProjectProtection` is set to `false`, which indicates that the project data protection mechanism is disabled. Users who are authorized to access multiple projects can perform cross-project data access operations to transfer data. If a project stores highly-sensitive data, the project administrator must configure the project data protection mechanism.

1.20.6.3. Data export methods after project data protection is enabled

This topic describes the data export methods that you can use after project data protection is enabled for MaxCompute.

After data protection is enabled for your project, you may soon encounter this situation: Alice submits a request to export the data of a table from the project. You approve the request from Alice because this table does not contain sensitive data. To meet the business requirements of Alice, MaxCompute provides two methods to export data after project data protection is enabled.

Configure an exception policy

If a project owner enables project data protection, the owner can run the following command to configure an exception policy:

```
set ProjectProtection=true with exception <policyFile>
```

Note The exception policy is different from the policy configured for policy-based authorization even though the two policies have the same syntax. This exception policy acts as an exception to the rules of project data protection. After the exception policy is configured, the access requests that meet the description of the exception policy are ignored by the project data protection rules.

Example of an exception policy:

The following policy allows Alice with the Alibaba Cloud account `Alice@aliyun.com` to export data to the `alipay` project when Alice uses SQL tasks to perform `SELECT` operations on the `alipay.table_test` table.

```
{
  "Version": "1", "Statement": [{
    "Effect": "Allow", "Principal": "ALIYUN$Alice@aliyun.com",
    "Action": ["odps:Select"],
    "Resource": "acs:odps:*:projects/alipay/tables/table_test",
    "Condition": {
      "StringEquals": { "odps:TaskType": ["DT", "SQL"]
    }
  }
}]
}
```

Note

- The exception policy is not a common authorization method. If Alice does not have the SELECT permission on the alipay.table_test table, Alice cannot export data even if the preceding exception policy is configured. Project data protection is a method to control data transfer but does not control data access. Project data protection is effective only if users can access their destination data.
- After an exception policy is configured, data leaks may occur due to time-of-check to time-of-use (TOCTOU). TOCTOU is also considered a race condition. Symptom:
 - i. [TOC stage] User A submits an application to the project owner to export the t1 table. After the owner verifies that the t1 table does not contain sensitive data, the owner configures an exception policy to authorize user A to export the t1 table.
 - ii. A malicious user writes sensitive data to the t1 table.
 - iii. [TOU stage] User A exports the t1 table. The t1 table exported by user A is not the table verified by the project owner.

To prevent this issue, the project owner must make sure that all project members, including the users who are assigned the admin role, do not have the UPDATE permission or the DROP and CREATE TABLE permissions on the table that a user wants to export. In the preceding example, we recommend that the project owner create a snapshot of the t1 table in Step i, and then use this snapshot to configure an exception policy. In addition, the project owner does not assign the admin role to other project members.

Configure a trusted project

If data protection is enabled for a project, data of the project can be exported to its trusted project. This export does not violate the rules of project data protection. If multiple projects are configured as trusted projects for each other, they form a trusted project group. Data of a project can be exported to other projects in the group but cannot be exported to the projects out of the group.

You can run the following commands to manage the trusted projects:

```
list trustedprojects
-- List all trusted projects that are configured for a project.
add trustedproject <projectname>
-- Add a project as a trusted project for the project.
remove trustedproject <projectname>
-- Remove a trusted project from the project.
```

1.20.6.4. Package-based resource sharing and project data protection

This topic describes the relationship between package-based resource sharing and data protection in MaxCompute projects.

In MaxCompute, package-based resource sharing and project data protection are mutually independent mechanisms that take effect at the same time, but their features are mutually exclusive.

In MaxCompute, package-based resource sharing takes precedence over project data protection. If a data object in a project is shared with users in other projects based on resource sharing, the project data protection rules will not apply to this object.

To prevent data transfer from a project to another, you must check the following items after you set ProjectProtection to True:

- Make sure that no trusted projects are added. If trusted projects are added, evaluate possible risks.
- Make sure that no exception policies are configured. If exception policies are configured, evaluate possible risks, especially the risks caused by time-of-check to time-of-use (TOC2TOU).
- Check whether package-based data sharing is not in use. If package-based data sharing is in use, make sure that the package contains no sensitive data.

1.20.7. Project security configurations

This topic describes security configurations for MaxCompute projects.

MaxCompute is a platform that allows multiple tenants to process data at the same time. When these tenants process data on MaxCompute, they may have different requirements for data security. To meet their requirements, MaxCompute allows tenants to configure security settings for projects. Project owners can specify an external account system that is supported by MaxCompute and an authentication model to suit their needs.

MaxCompute supports multiple orthogonal authorization mechanisms, such as ACL-based authorization, policy-based authorization, and implicit authorization. If implicit authorization is used, an object creator is automatically granted the permissions to access the object. However, not all users require these security mechanisms. You can configure an authentication model based on your business requirements and usage habits.

```
show SecurityConfiguration
-- View the security configuration of a project.
set security.CheckPermissionUsingACL=true/false
-- Enable or disable ACL-based authorization. The default value of security.CheckPermissionUsingACL is true.
set security.CheckPermissionUsingPolicy=true/false
-- Enable or disable policy-based authorization. The default value of security.CheckPermissionUsingPolicy is true.
set security.ObjectCreatorHasAccessPermission=true/false
-- Allow or disallow an object creator to be granted the object access permission by default. The default value of security.ObjectCreatorHasAccessPermission is true.
set security.ObjectCreatorHasGrantPermission=true/false
-- Allow or disallow an object creator to be granted the authorization permission by default. The default value of security.ObjectCreatorHasGrantPermission is true.
set security.LabelSecurity=true/false
-- Enable or disable the label security policy.
set security.ProjectProtection=true/false
-- Enable or disable project data protection to allow or disallow data transfer to other projects.
```

1.20.8. Authorization policies

1.20.8.1. Policy overview

This topic describes the policies that are used for policy-based authorization.

Policy-based authorization is based on principal. A principal can be a user or role. Policies are considered resources of a principal. You can perform policy-based authorization only on a principal that exists. If a principal is deleted, the policies of the principal are automatically deleted. Policy-based authorization uses a custom policy language defined by MaxCompute to perform authorization operations on principals. After the authorization is complete, principals are allowed or not allowed to access resources in projects.

Policy-based authorization is a new authorization mechanism. It is used to handle complicated scenarios in which ACL-based authorization struggles to deal with, such as:

- A principal is granted permissions to access a group of resources, such as all functions and all tables whose name starts with taobao, at a time.
- A principal is granted permissions with specified conditions, for example, the permission on a table is effective at a specified time, for a request from a specified IP address, or for only SQL tasks (not supported for other types of tasks)

Command syntax:

```
GET POLICY;
-- Read the policies of a project.
PUT POLICY <policyFile>;
-- Set a policy for the project and use this policy to overwrite an existing policy.
GET POLICY ON ROLE <roleName>;
-- Read the policies of a role in the project.
PUT POLICY <policyFile> ON ROLE <roleName>;
-- Set a policy for a role in the project and use this policy to overwrite an existing policy.
```

 **Note** MaxCompute supports project policies and role policies. A project policy applies to all users of the project, whereas a role policy applies only to users who are assigned roles. You must specify a user principal for project policies. You cannot specify a user principal for role policies because the principal has been assigned a role.

Example of configuring a project policy for policy-based authorization:

Scenario: A project policy needs to be configured to ensure that the user with the Alibaba Cloud account `alice@aliyun.com` can only submit a request from the CIDR block `10.32.180.0/23` before `2019-11-11 23:59:59`. The user can perform only the operations to create instances and tables and view the operations that are performed on the `test_project` project. The user is not allowed to delete tables from the `test_project` project.

The following project policy is configured:

```
{
  "Version": "1", "Statement": [{
    "Effect": "Allow", "Principal": "ALIYUN$alice@aliyun.com",
    "Action": ["odps:CreateTable", "odps:CreateInstance", "odps:List"],
    "Resource": "acs:odps:*:projects/test_project",
    "Condition": { "DateLessThan": {
      "acs:CurrentTime": "2017-11-11T23:59:59Z"
    }
  },
  "IpAddress": { "acs:SourceIp": "10.32.180.0/23"
  }
  },
  {
    "Effect": "Deny", "Principal": "ALIYUN$alice@aliyun.com", "Action": "odps:Drop", "Resource": "acs:odps:*:projects/test_project/tables/*"
  }
  ]
}
```

```.json

```

```json
{
  "Version": "1", "Statement": [{
    "Effect": "Allow", "Principal": "ALIYUN$alice@aliyun.com",
    "Action": ["odps:CreateTable", "odps:CreateInstance", "odps:List"],
    "Resource": "acs:odps:*:projects/test_project",
    "Condition": { "DateLessThan": {
      "acs:CurrentTime": "2017-11-11T23:59:59Z"
    }
  },
  "IpAddress": { "acs:SourceIp": "10.32.180.0/23"
}
}
},
{
  "Effect": "Deny", "Principal": "ALIYUN$alice@aliyun.com",
  "Action": "odps:Drop",
  "Resource": "acs:odps:*:projects/test_project/tables/*"
}]
}

```

```

{
  "Version": "1", "Statement": [{
    "Effect": "Allow", "Principal": "ALIYUN$alice@aliyun.com",
    "Action": ["odps:CreateTable", "odps:CreateInstance", "odps:List"],
    "Resource": "acs:odps:*:projects/test_project",
    "Condition": { "DateLessThan": {
      "acs:CurrentTime": "2017-11-11T23:59:59Z"
    }
  },
  "IpAddress": { "acs:SourceIp": "10.32.180.0/23"
}
}
},
{
  "Effect": "Deny", "Principal": "ALIYUN$alice@aliyun.com",
  "Action": "odps:Drop",
  "Resource": "acs:odps:*:projects/test_project/tables/*"
}]
}

```

Note

- Only role policies and project policies are supported. User policies are not supported.
- Only one policy file can be configured for each type of policy. When the PUT POLICY command is executed to configure a policy, the configured policy overwrites an existing policy. To modify a policy, perform the following steps:
 - i. Run the GET POLICY command.
 - ii. Manually merge policy statements.
 - iii. Run the PUT POLICY command.

1.20.8.2. Policy-related terminology

This topic describes the basic terms used in a policy.

Permission is a basic concept of access control. If a requester wants to take an action on a resource, the action may be allowed or denied, based on the permission settings. A statement refers to the formal description of a single permission. A policy refers to a set of statements.

An access policy consists of the following access control elements: principal, action, resource, access restriction, and effect. The following sections provide brief descriptions of these elements.

Principal

A principal of an object is a user or role to which permissions are assigned in an access policy. For example, the access policy allows Michael to perform the CreateObject action on the resource SampleBucket before December 31, 2019. Michael is the principal of the object.

Action

An action is an activity that the principal has permissions to perform. For example, the access policy allows Michael to perform the CreateObject action on the resource SampleBucket before December 31, 2019. CreateObject is an action of the access policy.

Resource

A resource is the object that a principal requests access to. For example, the access policy allows Michael to perform the CreateObject action on the resource SampleBucket before December 31, 2019. SampleBucket is a resource of the access policy.

Access restriction

Access restriction is the condition that specifies whether a permission takes effect. For example, the access policy allows Michael to perform the CreateObject action on resource SampleBucket before December 31, 2019. The access restriction is before December 31, 2019.

Effect

Authorization effect has two options: allow or deny. Deny takes precedence over allow during permission checks.

 **Notice** Deny and revocation of authorization are different. Revocation of authorization involves revocation of allow and deny. For example, conventional databases support the Revoke and Revoke Deny actions.

1.20.8.3. Access policy structure

1.20.8.3.1. Overview

This topic describes the language structure of policies defined for policy-based authorization.

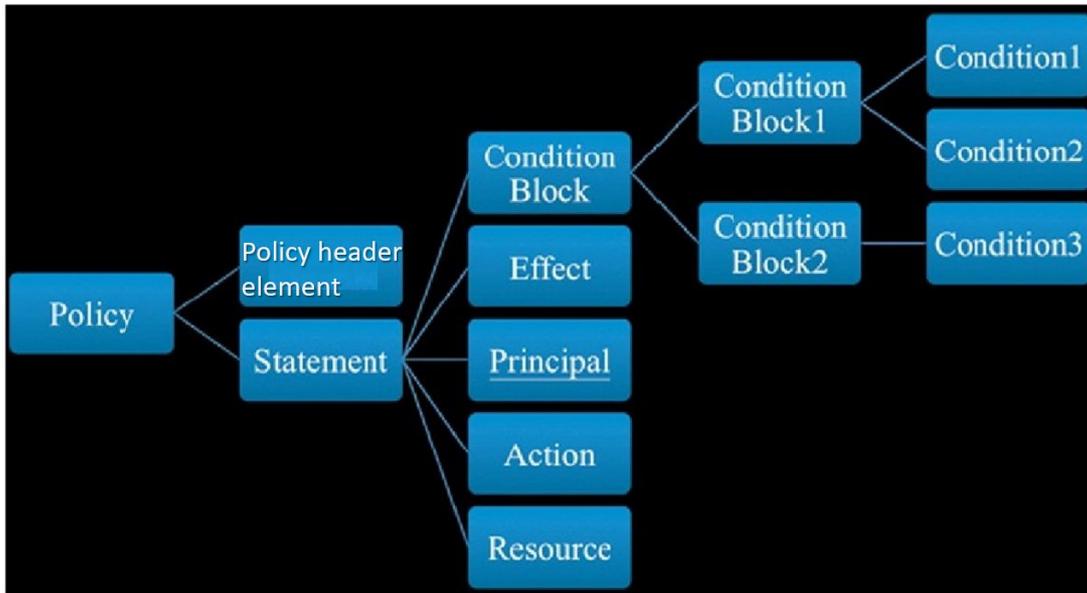
A policy consists of the following parts:

- An optional policy header
- One or more statements

 **Note** The policy header is optional and includes the policy version. The policy body is a set of statements.

The following figure shows the structure of a policy.

Structure of a policy



1.20.8.3.2. Authorization statement structure

This section describes the structure of the authorization statement in policies defined for policy-based authorization.

An authorization statement includes the following entries:

- **Effect**: indicates the permission type of this statement. The value can be Allow or Deny.
- **Principal**: If a policy is bound to a user or role in the authorization process, such as the role policy of MaxCompute, you cannot specify a principal. If a policy is bound to a project or objects of the project in the authorization process, such as the project policy of MaxCompute, you must specify a principal.
- **Action**: indicates the authorization operation. It can be one or multiple operation names and supports the asterisk (*) and question mark (?). For example, `Action = *` indicates all operations.
- **Resource**: indicates the authorization object. It can be one or multiple object names and supports the asterisk (*) and question mark (?). For example, `Resource = *` indicates all objects.
- **Condition block**: indicates the conditions that must be met for the permission described by this authorization statement to take effect.

1.20.8.3.3. Structure of condition blocks

This topic describes the structure of condition blocks in the statements that are used for policy-based authorization.

A condition block consists of one or more condition clauses. A condition clause consists of a condition action type, condition key, and condition value. The condition action types and condition keys are described in detail in subsequent topics.

The following rules are used to check whether conditions in a condition block are met:

- A condition key can correspond to one or more values. If the value of the condition key is equal to one of the specified values, the condition is met.
- If multiple condition keys in a clause of a condition action type are met, this clause is met.
- If condition keys in all condition clauses of a condition block are met, this condition block is met.

1.20.8.3.4. Condition action types

This topic describes the types of condition actions in policy condition clauses.

The actions on the following data types are supported: STRING, BOOLEAN, and IP address as well as the numeric, date, and time types. Methods supported by each action type:

STRING

StringEquals
StringNotEquals
StringEqualsIgnoreCase
StringNotEqualsIgnoreCase
StringLike
StringNotLike

Numeric

NumericEquals
NumericNotEquals
NumericLessThan
NumericLessThanEquals
NumericGreaterThan
NumericGreaterThanEquals

Date and Time

DateEquals
DateNotEquals
DateLessThan
DateLessThanEquals
DateGreaterThan
DateGreaterThanEquals

BOOLEAN

Bool

IP Address

IpAddress NotIpAddress

1.20.8.3.5. Condition keys

This topic describes the condition keys in policy condition clauses.

MaxCompute supports the condition keys reserved by Alibaba Cloud Service (ACS). The following table describes these condition keywords.

Condition keys

Condition key reserved by ACS	Data type	Description
-------------------------------	-----------	-------------

Condition key reserved by ACS	Data type	Description
<code>acs:CurrentTime</code>	Date and Time	The time when the web server receives a request. The time is based on the ISO 8601 standard, for example, 2019-11-11T23:59:59Z.
<code>acs:SecureTransport</code>	BOOLEAN	Whether the request is sent over a secure channel, such as an HTTPS channel.
<code>acs:SourceIp</code>	IP Address	The IP address of the client that sends a request.
<code>acs:UserAgent</code>	STRING	The User-Agent header in a request.
<code>acs:Referer</code>	STRING	The HTTP referer in a request.

 **Note** `acs:SourceIp` indicates the remote IP address of an HTTP connection. This condition key does not indicate an IP address in the IP address list represented by `x-forwarded-for` in the HTTP header. For example, if 10.230.205.105 is a LAN IP address, `acs:SourceIp` indicates the egress gateway IP address of this LAN. If the network egress uses a proxy server, `acs:SourceIp` indicates the IP address of the proxy server. If the request traverses across multiple proxy servers, `acs:SourceIp` indicates the IP address of the final proxy server. The value of `acs:SourceIp` may vary based on the rules configured on proxy servers.

1.20.8.4. Access policy norm

1.20.8.4.1. Principal naming conventions

This topic describes the principal naming conventions in language specifications of policies defined for policy-based authorization.

The principal is the request sender. Only an Alibaba Cloud account, domain account, or Taobao account can be used as a principal. An Alibaba Cloud account can be represented by ID or `DisplayName`.

Example:

```
"Principal": "43274"
"Principal": "ALIYUN$bob@aliyun.com"
"Principal": ["ALIYUN$bob@aliyun.com", "ALIYUN$jack@aliyun.com", "TAOBAO$alice"]
```

1.20.8.4.2. Naming conventions of resources

This topic describes the naming conventions of resources in language specifications of policies defined for policy-based authorization.

MaxCompute resources are named in the following format:

```
acs:<service-name>:<namespace>:<relative-id>
```

The following table describes the parameters in this format.

Parameters

Parameter	Description
<code>acs</code>	The retained resource header.

Parameter	Description
service-name	The name of an Alibaba Cloud service (ACS), such as MaxCompute, Object Storage Service (OSS), and Tablestore.
namespace	The namespace that is used for resource isolation. If an Alibaba Cloud account is used for resource isolation, the value can be an Alibaba Cloud account. If this parameter is not required, you can use an asterisk (*) instead.
relative-id	The service-related resource. Its meaning is based on services. The value format of this parameter supports a tree structure that is similar to the file path. The following value format of relative-id is used for MaxCompute: <div style="background-color: #f0f0f0; padding: 5px; margin-top: 5px;"> projects/<project_name>/<object_type>/<object_name> </div>

The following table provides examples of MaxCompute resource names.

Examples of MaxCompute resource names

Name	Description
*	Indicates all objects in the project.
projects/prj1/tables/t1	Indicates the t1 table of the prj1 project.
projects/prj1/instances/*	Indicates all instances in the prj1 project.
projects/prj1/tables/*	Indicates all tables in the prj1 project.
projects/prj1/tables/taobao*	Indicates all tables in the prj1 project. The table names are all prefixed by taobao.

1.20.8.4.3. Naming conventions of actions

This topic describes the naming conventions of actions in language specifications of policies defined for policy-based authorization.

Naming convention of actions:

```
<service-name>:<action-name>
```

Parameters:

- **service-name**: the name of an Alibaba Cloud service, for example, MaxCompute, Object Storage Service (OSS), and Tablestore.
- **action-name**: the name of the service-related action.

The following table provides examples of MaxCompute action names.

Examples of MaxCompute action names

Action name	Description
*	Indicates all actions.
odps:*	Indicates all MaxCompute actions.

Action name	Description
<code>odps:CreateTable</code>	Indicates the CreateTable action of MaxCompute.
<code>odps:Create*</code>	Indicates all MaxCompute actions whose names start with Create.

1.20.8.4.4. Naming conventions of condition keys

This topic describes the naming conventions of condition keys in language specifications of policies defined for policy-based authorization.

Condition keys that are reserved for Alibaba Cloud Services (ACSs) are named in the following format:

```
acs:<condition-key>
```

Parameter: condition-key: the condition key, which is accessible to all ACSs. The following condition keys are supported: `acs:CurrentTime`, `acs:SecureTransport`, `acs:SourceIp`, `acs:UserAgent`, and `acs:Referer`.

Condition keys that are specific to an ACS are named in the following format:

```
<service-name>:<condition-key>
```

Parameter: condition-key: the condition key that is specific to an ACS.

1.20.8.4.5. Example of a policy for policy-based authorization

This topic provides an example of a policy for policy-based authorization.

Sample policy:

```
{
  "Version": "1",
  "Statement": [{
    "Effect": "Allow",
    "Principal": "ALIYUN$alice@aliyun.com",
    "Action": ["odps:CreateTable", "odps:CreateInstance", "odps:List"],
    "Resource": "acs:odps:*:projects/prj1",
    "Condition": { "DateLessThan": {
      "acs:CurrentTime": "2019-11-11T23:59:59Z"
    }
    },
    "IpAddress": { "acs:SourceIp": "10.32.180.0/23"
    }
  }
  ],
  {
    "Effect": "Deny", "Principal": "ALIYUN$alice@aliyun.com",
    "Action": "odps:Drop",
    "Resource": "acs:odps:*:projects/prj1/tables/*"
  }
}
```

 **Note** The user with the Alibaba Cloud account `alice@aliyun.com` is authorized to submit a request from CIDR block `10.32.180.0/23` only before `2019-11-11T23:59:59Z`. The user has only the `CREATE INSTANCE`, `CREATE TABLE`, and `LIST` permissions on the `prj1` project. The user is not allowed to delete tables from the `prj1` project.

1.20.8.5. Differences between policy-based authorization and ACL-based authorization

This topic describes the differences between policy-based authorization and ACL-based authorization.

ACL-based authorization

- Before you grant or revoke permissions, make sure that the grantee, such as a user or role and an object such as a table exists. This also applies to authorization in Oracle and the purpose is to avoid the security risks that may occur after you delete and recreate an object with the same name.
- If you delete an object, all permissions related to the object are automatically revoked.
- Only allow (whitelist) authorization is supported. Deny (blacklist) authorization is not supported.
- The `GRANT` and `REVOKE` commands are used for ACL-based authorization. These commands are easy to use and not prone to mistakes. Conditional authorization is not supported.
- ACL-based authorization is suitable for simple scenarios where conditional authorization or a deny action is not required, and only the existing objects need to be authorized.

Policy-based authorization

- Before you grant or revoke permissions, you are not required to check whether the grantees or object exists. You can use double quotation marks "" to represent objects. If you use `projects/tbproj/tables/taobao` to represent an object, the object is all tables whose names start with `taobao` in the `tbproj` project. Similar to authorization in MySQL, policy-based authorization allows you to grant permissions to a non-existent object. The grantee must consider the security risks that may occur after you delete and recreate an object with the same name.
- If you delete an object, all permissions related to the object are not revoked.
- Both allow (whitelist) authorization and deny (blacklist) authorization are supported. If allow (whitelist) authorization and deny (blacklist) authorization are performed at the same time, the deny (blacklist) authorization takes precedence.
- Conditional authorization is supported. The grantee can enforce a maximum of 20 conditions on allow or deny authorization. For example, these conditions can be used to specify a valid range of IP addresses for requestors and the access time that must be earlier than the specified time.
- Policy-based authorization is suitable for relatively complicated scenarios where conditional authorization and deny actions are required for non-existent objects.
- The commands for policy-based authorization are used and these commands are complex.

1.20.8.6. Limits

This topic describes the limits of policy-based authorization.

Limits

Item	Upper Limit	Description
------	-------------	-------------

Item	Upper Limit	Description
ACCESS_POLICY_SIZE_LIMIT	32 KB	The maximum size of the text of a policy.
USER_NUMBER_LIMIT_IN_ONE_PROJECT	1000	The maximum number of users that can be added to a project.
ROLE_NUMBER_LIMIT_IN_ONE_PROJECT	500	The maximum number of roles that can be created for a project.
ROLE_NAME_LENGTH_LIMIT	64 KB	The maximum character length of a role name.
SECURITY_COMMENT_SIZE_LIMIT	1 KB	The maximum character length of a comment.
PACKAGE_NAME_LENGTH_LIMIT	128 KB	The maximum character length of a package name.
ALLOW_PROJECT_NUMBER_LIMIT_IN_ONE_PACKAGE	1024	The maximum number of projects for which a package can be installed.
RESOURCE_NUMBER_LIMIT_IN_ONE_PACKAGE	256	The maximum number of resources that can be included in a package.
PACKAGE_NUMBER_LIMIT_IN_ONE_PROJECT	512	The maximum number of packages that can be created for a project.
INSTALLED_PACKAGE_NUMBER_LIMIT_IN_ONE_PROJECT	64	The maximum number of packages that can be installed for a project.

1.20.9. Collection of security statements

1.20.9.1. Project security configurations

This topic describes the statements that are used for security configurations of projects.

Authentication configurations

Statements for authentication configurations

Statement	Description
show SecurityConfiguration	Allows you to view security configurations of a project.
set CheckPermissionUsingACL=true/false	Allows you to enable or disable ACL-based authorization.
set CheckPermissionUsingPolicy=true/false	Allows you to enable or disable policy-based authorization.
set ObjectCreatorHasAccessPermission=true/false	Allows you to grant or revoke default access permissions to or from object creators.

Statement	Description
<code>set ObjectCreatorHasGrantPermission=true/false</code>	Allows or disallows an object creator to be granted the ACL-based authorization permission by default.

Project data protection

Statements for project data protection

Statement	Description
<code>set ProjectProtection=false</code>	Allows you to disable project data protection.
<code>set ProjectProtection=true [with exception <policy>]</code>	Allows you to enable project data protection.
<code>list TrustedProjects</code>	Allows you to view trusted projects.
<code>add TrustedProject <projectName></code>	Allows you to add a project to trusted projects.
<code>remove TrustedProject <projectName></code>	Allows you to remove a project from trusted projects.

1.20.9.2. Project permission management

This topic describes the statements used for permission management of projects.

User management

Statements

Statement	Description
<code>list users</code>	Allows you to view all users that are added to a project.
<code>add user <username></code>	Allows you to add a user.
<code>remove user <username></code>	Allows you to remove a user.

Role management

Statements

Statement	Description
<code>list roles</code>	Allows you to view all the existing roles.
<code>create role <rolename></code>	Allows you to create a role.
<code>drop role <rolename></code>	Allows you to delete a role.
<code>grant <rolelist> to <username></code>	Allows you to revoke roles from a user.
<code>revoke <rolelist> from <username></code>	Allows you to assign one or multiple roles to a user.

ACL-based authorization

Statements

Statement	Description
grant <privList> on <objType> <objName> to user <username>	Allows you to grant permissions to a user
grant <privList> on <objType> <objName> to role <rolename>	Allows you to grant permissions to a role.
revoke <privList> on <objType> <objName> from user <username>	Allows you to revoke permissions from a user.
revoke <privList> on <objType> <objName> from role <rolename>	Allows you to revoke permissions from a role.

Policy-based authorization

Statements

Statement	Description
get policy	Allows you to view policy settings of a project.
put policy <policyFile>	Allows you to configure a policy for a project
get policy on role <roleName>	Allows you to view the policy settings of a role.
put policy <policyFile> on role <roleName>	Allows you to configure a policy for a role.

Permission review

Statements

Statement	Description
whoami	Allows you to view the information about a user.
show grants [for <username>] [on type <objectType>]	Allows you to view the permissions and roles of a user.
show acl for <objectName> [on type <objectType>]	Allows you to view the authorization information of an object.
describe role <roleName>	Allows you to view the authorization and assignment information of a role.

1.20.9.3. Package-based resource sharing

This topic describes the statements that are used for package-based resource sharing.

Share resources

Statements for sharing resources

Statement	Description
<code>create package <pkgName></code>	Allows you to create a package.
<code>delete package <pkgName></code>	Allows you to delete a package.
<code>add <objType> <objName> to package<pkgName> [with privileges privs]</code>	Allows you to add resources that you want to share to a package.
<code>remove <objType> <objName> from package <pkgName></code>	Allows you to remove shared resources from a package.
<code>allow project <prjName> to install package <pkgName> [using label <num>]</code>	Allows a project to use a package of a specified user.
<code>disallow project <prjName> to install package <pkgName></code>	Disallows a project from using the package.

Use resources

Statements for using resources

Statement	Description
<code>install package <pkgName></code>	Allows you to install a package.
<code>uninstall package <pkgName></code>	Allows you to uninstall a package.

View packages

Statements for viewing packages

Statement	Description
<code>show packages</code>	Allows you to view all the packages that are created and installed.
<code>describe package <pkgName></code>	Allows you to view the details of a package.

1.21. Hierarchical throttling

1.21.1. Overview

MaxCompute provides user-based job throttling.

If some users perform inappropriate operations, a series of issues may arise. For example, if a user submits a large number of jobs, the jobs may be queued for a long period of time and resources in the resource group may be exhausted. To address these issues, MaxCompute provides user-based job throttling in this version and later. After user-based job throttling is enabled, the project owner can specify an upper limit for the number of jobs that can be concurrently submitted based on the resources required for the business or teams.

1.21.2. Hierarchical throttling

MaxCompute supports hierarchical throttling.

MaxCompute uses global throttling for system-wide configurations. To properly control project resources, MaxCompute also allows project owners to use hierarchical throttling for a project.

 **Note** Project owners can use hierarchical throttling rules to specify the maximum number of instances that are allowed for a project.

1.21.3. Specify the maximum number of instances that are allowed for a project

This topic describes how to use hierarchical throttling rules to specify the maximum number of instances that are allowed for a project.

Formulate JSON rules for throttling

Assume that the following throttling rules are defined:

- test_rule1: indicates that the maximum number of instances that users except user1 can run in the information_schema_dev project at the same time is two.
- test_rule2: indicates that the maximum number of instances that user1 can run in the information_schema_dev project at the same time is four.

If the user field is set to a specified user other than an asterisk (*), the priority of the specified user is higher than the priorities of other users. After the two rules take effect, users except user1 can run a maximum of two instances in the information_schema_dev project at the same time. However, user1 can run a maximum of four instances in the information_schema_dev project at the same time.

In addition, JSON rules of a project must be written in the same list.

Sample rules:

```
[{
  "name": "test_rule1",
  "entity": {
    "project": "information_schema_dev",
    "user": "*"
  },
  "settings": [{
    "metric": "total_instances",
    "threshold": "2",
    "action": "deny"
  }]
},
{
  "name": "test_rule2",
  "entity": {
    "project": "information_schema_dev",
    "user": "user1"
  },
  "settings": [{
    "metric": "total_instances",
    "threshold": "4",
    "action": "deny"
  }]
}]
```

 **Note** In the sample rules, you must specify name, project, user, and threshold. Other fields are reserved for future use and must retain their default values.

Configure and view throttling rules

Convert the preceding sample rules in the JSON format into a single row, and run the `setproject` command in the MaxCompute console to configure throttling rules.

```
information_schema_dev> setproject THROTTLING_RULES=[{"name":"test_rule1","entity":{"project":"information_schema_dev","user":"*"},"settings":[{"metric":"total_instances","threshold":"2","action":"deny"}]},{"name":"test_rule2","entity":{"project":"information_schema_dev","user":"19655xxxxx48481"},"settings":[{"metric":"total_instances","threshold":"4","action":"deny"}]}];
```

Run the following command to view the throttling rules:

```
information_schema_dev> setproject;
```

View the output information for throttling

If a large number of instances that meet the throttling rules run in the `information_schema_dev` project, an error is returned. Example of error information:

```
FAILED: Request rejected by flow control. Break rule: test_rule2. Matched instance count: 10. Limit: 4
-- The request is rejected because test_rule 2 is violated. The number of instances that match this rule is 10. However, this rule specifies that a maximum of 4 instances can run at the same time.
```

Delete throttling rules

You can leave `THROTTLING_RULES` empty to delete the throttling rules.

```
information_schema_dev> setproject THROTTLING_RULES=[];
```

Usage notes

For ease of description, rules can be written in the following format: `/project/user/empid? limit=x`. Wildcards are allowed in rules. If a rule with wildcards matches a throttling rule, a rule priority issue may occur.

In this case, the throttling rule has a higher priority than the rule with wildcards. The rule priority issue occurs only if a rule with wildcards matches a throttling rule. In other cases, the rule with wildcards and the throttling rule take effect at the same time.

Example 1: A throttling rule takes effect.

- Rule 1: `/meta_dev/*? limit=10`
- Rule 2: `/meta_dev/123456789? limit=100`

In this example, Rule 1 matches Rule 2. Rule 2 is a throttling rule and takes effect. Rule 1 is a rule with wildcards and does not take effect. The user with the ID of 123456789 in the `meta_dev` project can run a maximum of 100 instances at the same time. Other users can run a maximum of 10 instances at the same time.

Example 2: Two throttling rules take effect at the same time.

- Rule 3: `/meta_dev? limit=10`
- Rule 4: `/meta_dev/123456789? limit=100`

In this example, the two rules take effect at the same time. The two rules do not contain wildcards. Therefore, no rule priority issues occur. The user with the ID of 123456789 in the meta_dev project can run a maximum of 10 instances at the same time. This is because Rule 3 indicates that the total number of instances that all users in the meta_dev project can run at the same time is 10. This rule applies even if the user with the ID of 123456789 is allowed to run 100 instances at the same time.

Example 3: A throttling rule and a rule with wildcards take effect at the same time.

- Rule 5: /meta_dev/*? limit=10
- Rule 6: /meta_dev/123456789/111111? limit=100

In this example, the two rules take effect at the same time. Rule 5 is a rule with wildcards but does not match Rule 6. This is because a rule with wildcards matches only a throttling rule at the same directory level. Therefore, no rule priority issues occur.

1.22. Frequently-used tools

1.22.1. MaxCompute console

1.22.1.1. Usage notes

This topic provides usage notes for the MaxCompute client.

- Do not parse data based on the output format of the MaxCompute client. The client output format may not be forward compatible. The command syntax and behavior vary based on client versions.
- The MaxCompute client is a Java program. It requires JRE to run. You must download and install JRE 1.8 to run the MaxCompute client.
- Before you configure the client, make sure that a project is created by using an Alibaba Cloud account and the AccessKey ID and AccessKey secret of the account are obtained.
- For more information about how to use the MaxCompute client, see [Configure the client](#).

1.22.1.2. Install the MaxCompute client

This topic describes how to install the MaxCompute client.

1. Download the [client](#) package to your computer.
2. Decompress the package to the folder where you want to store the client. The package contains the following folders:

```
bin/
conf/
lib/
plugins/
```

3. Edit the following information in the odps_config.ini file in the conf folder:

```
project_name=my_project
access_id=*****
access_key=*****
end_point= <Endpoint of MaxCompute>
```

- Set access_id to the AccessKey ID of your Alibaba Cloud account and access_key to the AccessKey secret of your Alibaba Cloud account.
- project_name=my_project specifies the project that you want to access. This is the default project that you access each time you log on to the client. If project_name is not specified, you must run the `use project_name` command to access the project after you log on to the client.

- Set `end_point` to the endpoint of MaxCompute. The endpoint varies based on the user account.
4. After the modification, run `./bin/odpscmd` in the Linux operating system or `./bin/odpscmd.bat` in the Windows operating system to execute SQL statements. Example:

```
create table tbl1(id bigint);
insert overwrite table tbl1 select count(*) from tbl1;
select 'welcome to MaxCompute!' from tbl1;
```

1.22.1.3. Configuration-related operations

This topic describes the configuration operations and related parameters of the MaxCompute client.

View the help information

Run the following command to view the help information about the client:

```
odps@ > ./bin/odpscmd -h;
```

 **Note** You can also type `h;` or `help;` (not case-sensitive) in interactive mode.

Specify startup parameters

Run the following command to specify startup parameters:

```
Usage: odpscmd [OPTION]...
where options include:
--help (-h) for help
--project= use project
--endpoint= set endpoint
-u -p user name and password
-k will skip beginning queries and start from specified position
-r set retry times
-f <"file_path;"> execute command in file
-e <"command;[command;]..."> execute command, include sql command
-C will display job counters
```

Example of specifying the `-f` parameter

1. Prepare the `script.txt` file that is stored in `D:/`. This file contains the following content:

```
drop table if exists test_table_mj;
create table test_table_mj (id string, name string);
drop table test_table_mj;
```

2. Run the following command:

```
odpscmd\bin>odpscmd -f d:/script.txt;
```

3. View the output information. Sample output:

```

ID = 20170528122432906gux77io3
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-inc.com/api&podps_public_dev&i20170528122432906gux77io3&tokenRnlrSzJoL242YW43dFFlc1dmb1ZWZzFzQ1RFPSxPRFBTX09CTzoxMDcwMDI1Njl3ODA1Njl5LDE0MzM0MjA2NmMseyJTdGF0ZW1lbnQiOlt7IkFjdGlvbil6WyJvZHBzOjIjYXQxXSwiRWZmZWNOljojQWxsb3ciLCSZSNvdXJzSI6WyJhY3M6b2RwczoqOnB yb2plY3RzL29kcHNfcHVibGljX2Rld i9pbN0YW5jZXMvMjAxNTA1Mjgxmjl0MzI5MDZndXg3N2lvMyJdfV0sIlZlcnNpb24iOixln0=
OK
ID = 20170528122439318gcmkk6u1
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-inc.com/api&podps_public_dev&i20170528122439318gcmkk6u1&tokenSt0RXdlV0M5YjZET2I1MnJuUFkzWDN1aWpzPSxPRFBTX09CTzoxMDcwMDI1Njl3ODA1Njl5LDE0MzM0MjA2ODASEyJTdGF0ZW1lbnQiOlt7IkFjdGlvbil6WyJvZHBzOjIjYXQxXSwiRWZmZWNOljojQWxsb3ciLCSZSNvdXJzSI6WyJhY3M6b2RwczoqOnB yb2plY3RzL29kcHNfcHVibGljX2Rld i9pbN0YW5jZXMvMjAxNTA1Mjgxmjl0MzkzMT hnY21razZ1MSJdfV0sIlZlcnNpb24iOixln0=
OK
ID = 20170528122440389g98cmlmf
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-inc.com/api&podps_public_dev&i20170528122440389g98cmlmf&tokenNwLwL0EvQThxUXhzcTRERdc5NFg0b2lxZ3QwPSxPRFBTX09CTzoxMDcwMDI1Njl3OD A1Njl5LDE0MzM0MjA2ODASEyJTdGF0ZW1lbnQiOlt7IkFjdGlvbil6WyJvZHBzOjIjYXQxXSwiRWZmZWNOljojQWxsb3ciLCSZSNvdXJzSI6WyJhY3M6b2RwczoqOnB yb2plY3RzL29kcHNfcHVibGljX2Rld i9pbN0YW5jZXMvMjAxNTA1Mjgxmjl0NDazODlnOThjbWxtZiZdfV0sIlZlcnNpb24iOixln0=
OK

```

Enter the interactive mode

You can enter the interactive mode immediately after you run the following command on the MaxCompute client.

```

[admin: ~]$odpscmd
Aliyun ODPS Command Line Tool
Version 1.0
@Copyright 2012 Alibaba Cloud Computing Co., Ltd. All rights reserved.
XXX@ XXX> insert overwrite table dual select * from dual;

```

Note The first XXX indicates the identifier of MaxCompute, and the second XXX indicates the project to which you belong. Enter a command that is terminated by a semicolon (;) at the cursor, and press Enter to run the command.

Specify the output format

The output of an SQL statement is in the human-readable or machine-readable format. Human-readable is the default format. If you use the -M parameter when you run odpscmd, the output format is CSV.

Note This feature applies only to SELECT and READ statements and takes effect during data reading.

Specify the statements to execute

If you specify the `-e` or `-f` parameter and want to start with an intermediate statement among a few statements, you can specify the `-k` parameter. The `-k` parameter indicates that the execution starts from the specified statement and the preceding statements are skipped. If the value of the `-k` parameter is less than or equal to 0, the execution starts from the first statement. A statement terminated by a semicolon (`;`) is considered valid. At runtime, the output information indicates the specific statement that is executed or failed.

For example, the `dual.sql` file in the `tmp` folder contains the following SQL statements:

```
drop table dual;
create table dual (dummy string);
insert overwrite table dual select count(*) from dual;
```

You can run the following command to skip the first two statements:

```
odpscmd -k 3 -f dual.sql
```

Obtain information about the current logon user

Run a command to obtain the Alibaba Cloud account of the current logon user and the endpoint that is used. Command syntax:

```
whoami
```

Sample command:

```
odps@hiveut>whoami; Name: odptest@aliyun.com ID: 1090142773636588 End_Point: <Endpoint of MaxCompute > Project: lijunsecuritytest
```

Exit the MaxCompute client

Command syntax:

```
odps@ > quit;
```

You can also run the following command:

```
q;
```

Configure a job priority

Command syntax:

```
Admin@ > ./bin/odpscmd --instance-priority=<PRIORITY>;
```

Configuration file: `odps_config.ini`

```
instance_priority=<PRIORITY>
```

 Notice

- <PRIORITY >
- The priority setting in the configuration file applies to all instances submitted in the MaxCompute client.
- If you do not configure the priority for a job, its priority is 9.

Dry run mode

In dry run mode, MaxCompute parses an SQL statement to check whether the syntax of this statement is correct and generates an execution plan. In this mode, MaxCompute does not submit a distributed job. Command syntax:

```
./bin/odpscmd -y
```

SQL reliability

If you execute statements, such as INSERT or CREATE TABLE AS, on the MaxCompute client and an exception occurs when you run a job, the MaxCompute client automatically restores data and metadata to the status before the SQL query is run based on the known information. The exceptions include:

- Data that is overwritten by using the INSERT OVERWRITE statement during queries is restored from the temporary backup directory to the original directory.
- Data that is generated by using the INSERT INTO statement during queries is deleted.
- Tables that are created during queries or the dynamically generated partition information that does not exist before queries are deleted.

If MaxCompute fails to restore data, the following error code is returned. The error code informs you that further attempts may cause some data unrecoverable. Error:

```
ODPS-0110999: Critical! Internal error happened in commit operation and rollback failed, possible breach of atomicity
```

1.22.2. Eclipse development plugin

1.22.2.1. Install the Eclipse development plug-in

This topic describes how to install the Eclipse development plug-in.

Context

MaxCompute provides the Eclipse development plug-in to help you use SDKs for Java to develop MapReduce programs or user-defined functions (UDFs). This plug-in can simulate the running process of MapReduce programs or UDFs. It provides local debugging methods and features to generate templates. You can click [Eclipse](#) to download the plug-in package.

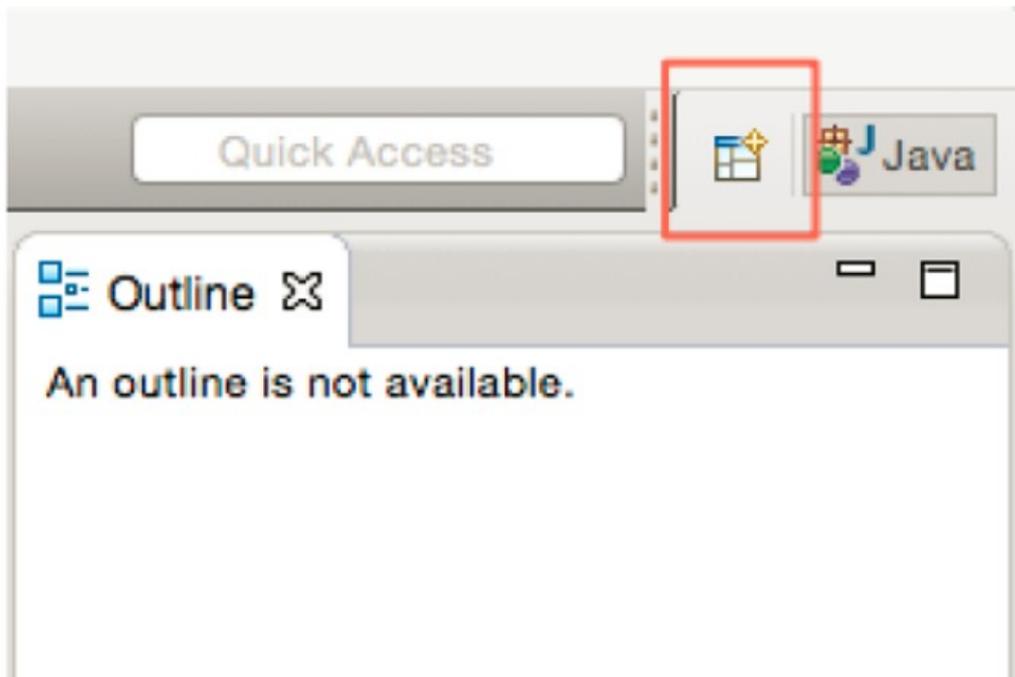
 **Note** Unlike the MapReduce program that runs in local mode, the Eclipse development plug-in cannot synchronize data with MaxCompute. You must manually copy the required data to the *warehouse* directory of the Eclipse development plug-in.

Procedure

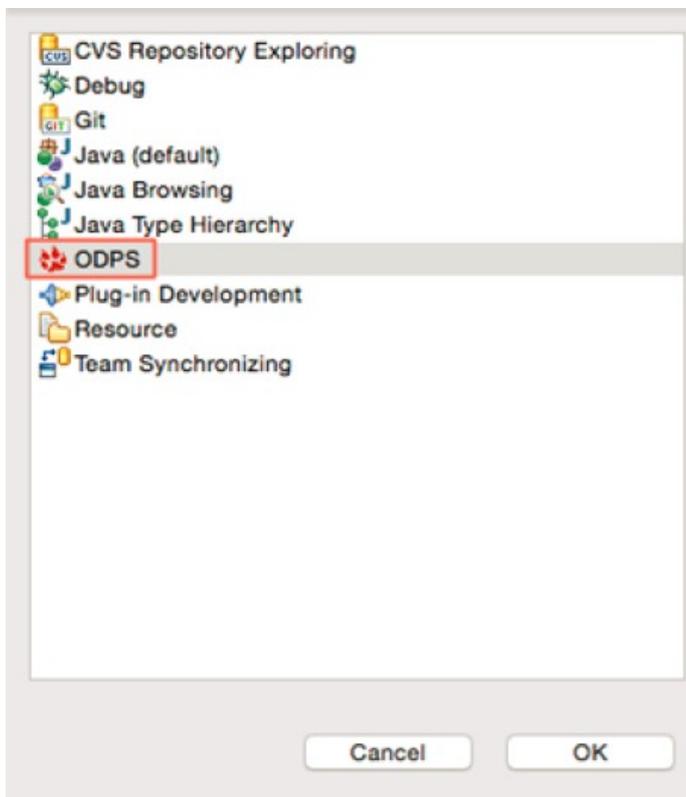
1. Decompress the Eclipse package to view the following JAR file in the package: *odps-eclipse-plugin-bundle*

-0.16.0.jar

2. Place the JAR file in the *plugins* folder under the Eclipse installation directory.
3. Start the Eclipse development plug-in and click the **Open Perspective** icon in the upper-right corner, as shown in the following figure.

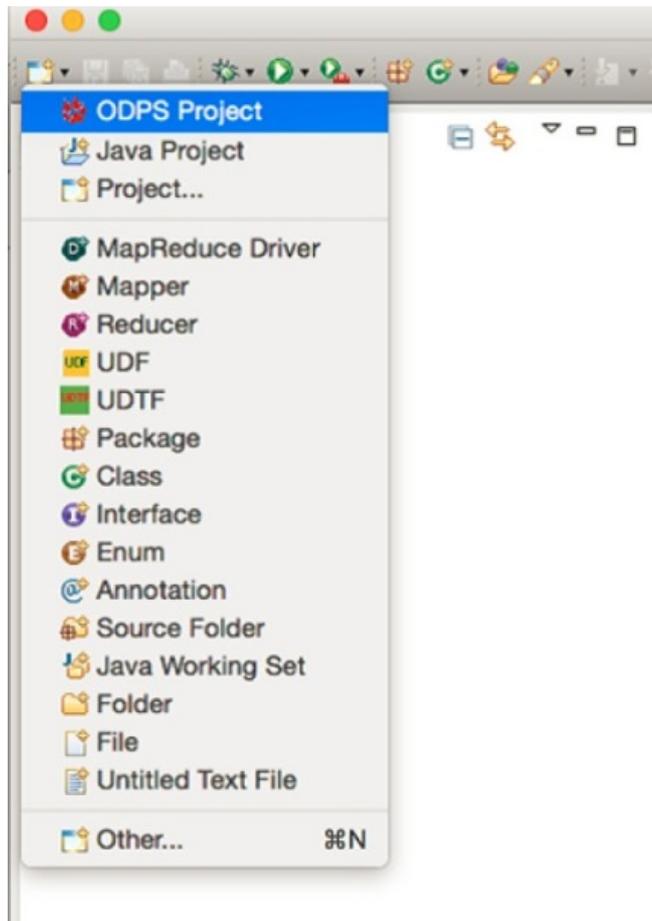


4. In the dialog box that appears, click **ODPS** and then **OK**.



5. In the navigation bar that appears, select the **ODPS project** and click **OK**. The **ODPS icon** is displayed in

the upper-right corner, as shown in the following figure. The icon indicates that the plug-in has taken effect.



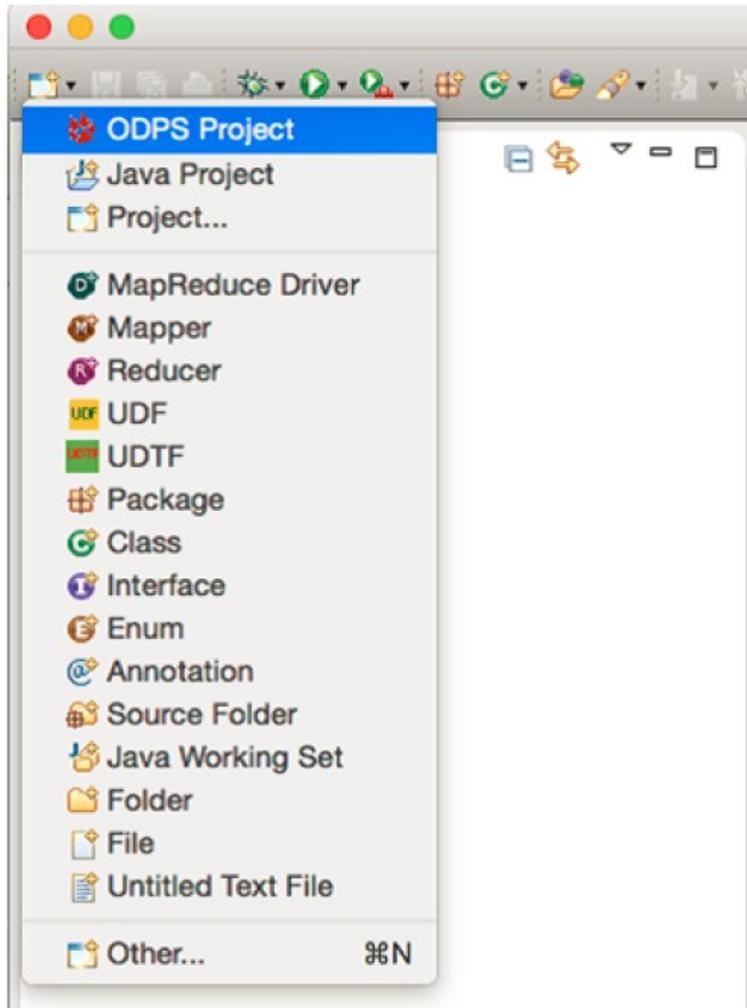
1.22.2.2. Create a project

1.22.2.2.1. Method 1

This topic describes Method 1. This method is used to create a project in the Eclipse development plug-in.

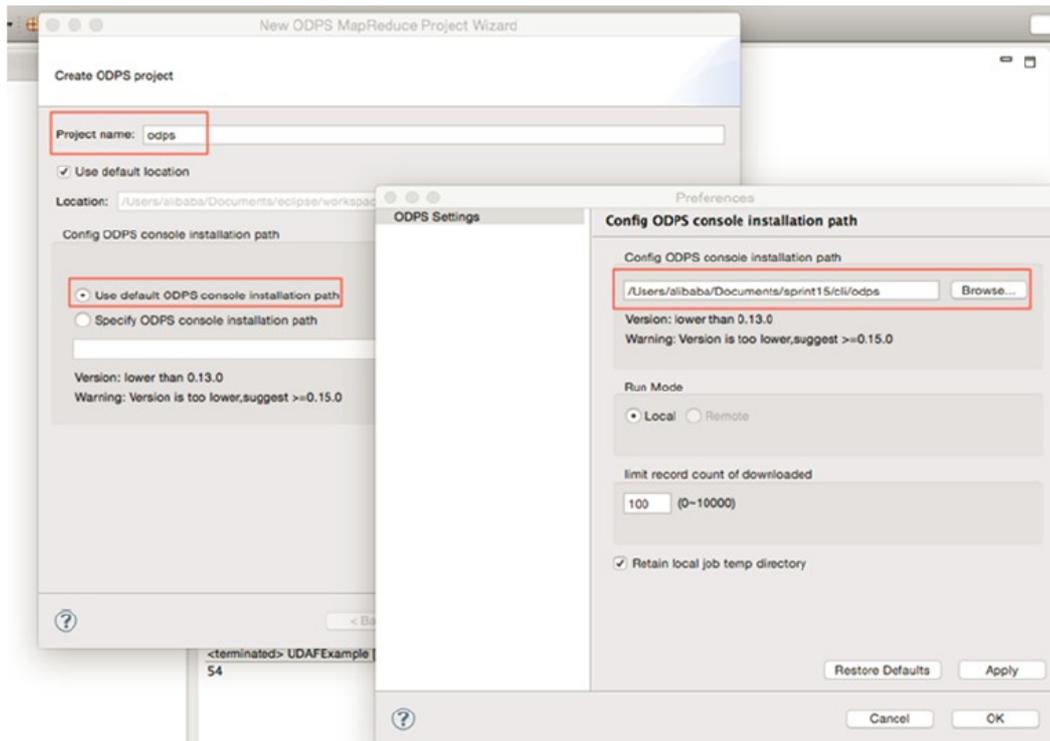
Procedure

1. In the upper-left corner of Eclipse, choose **File > New > Project > ODPS > ODPS Project**, as shown in the following figure.



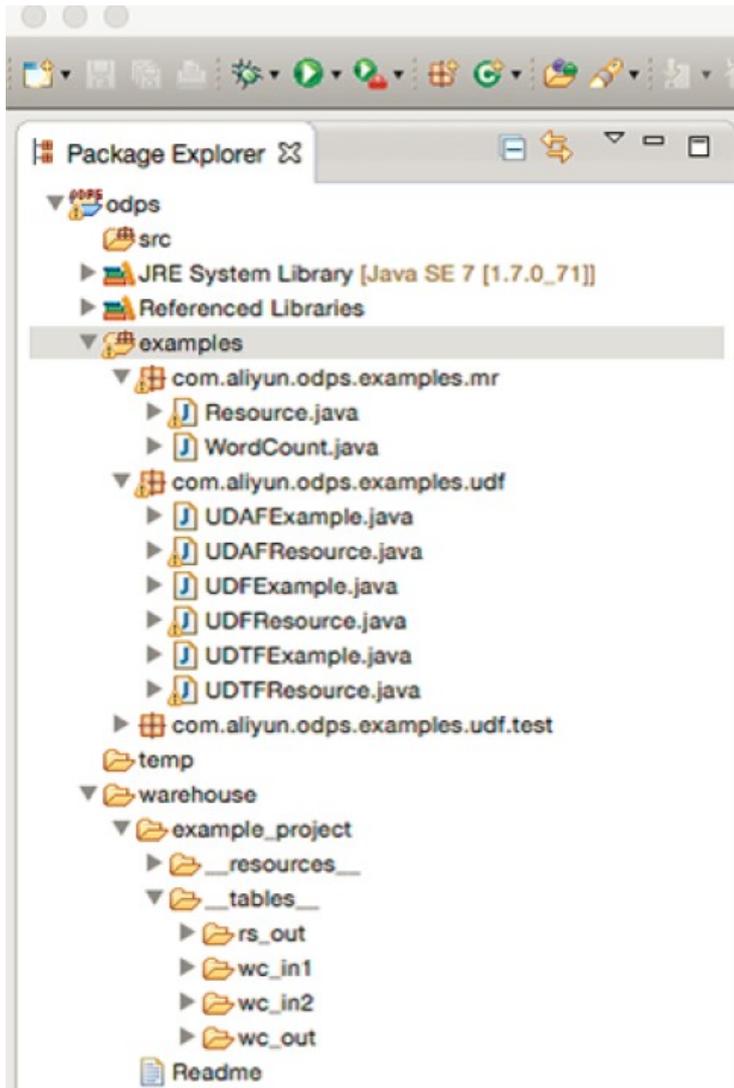
 Note In this example, the project name is ODPS.

2. In the dialog box shown in the following figure, enter the project name, select the installation path of the MaxCompute client, and then click **Finish**.



 **Note** The client must be installed in advance.

3. After you create a project, you can view the directory structure on the Package Explorer tab, as shown in the following figure.

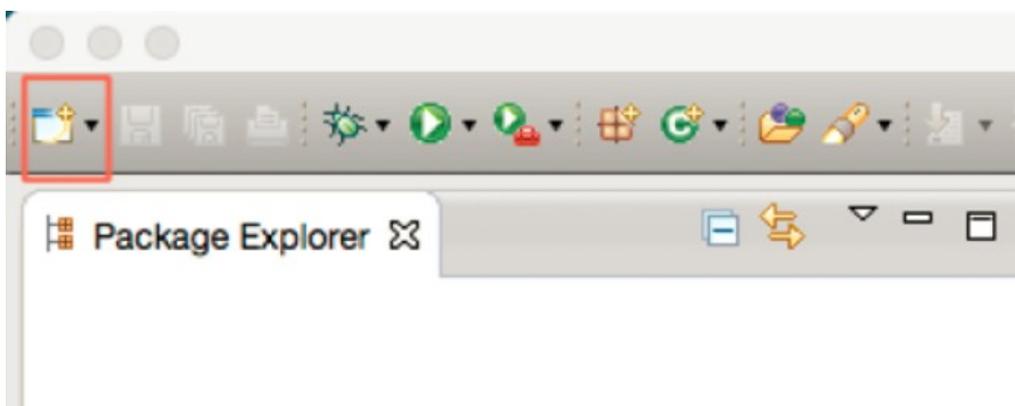


1.22.2.2.2. Method 2

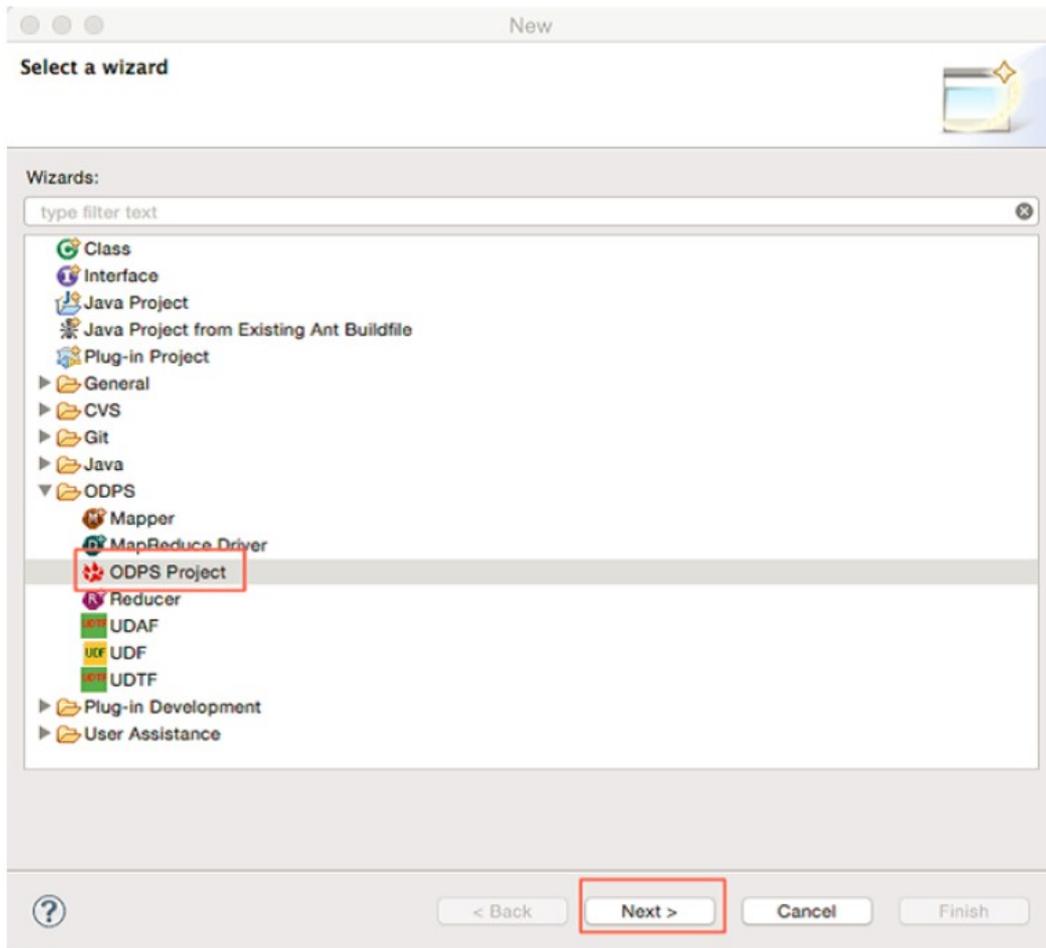
This topic describes another method to create a project by using the Eclipse plug-in.

Procedure

1. In the toolbar of the Eclipse plug-in, click the **New** icon, as shown in the following figure.



2. In the dialog box that appears, unfold ODPS, click **ODPS Project**, and then click **Next**, as shown in the following figure.



 **Note** In this example, the project name is ODPS.

3. The subsequent steps are the same as those in method 1. After you install the Eclipse plug-in, you can use it to compile a MapReduce program or user-defined function (UDF).

 **Note** For more information about how to run a MapReduce program by using the Eclipse plug-in, see [MapReduce running example](#). For more information about how to develop and run a UDF, see [UDF development and running example](#).

1.22.2.3. MapReduce running example

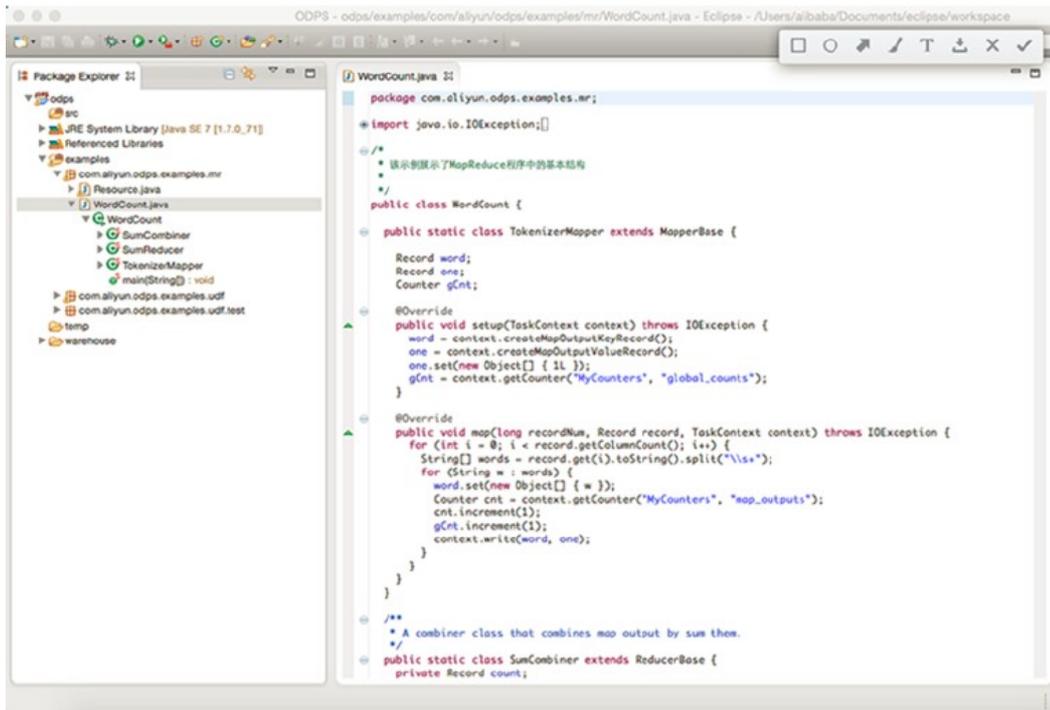
1.22.2.3.1. Run a WordCount program

This topic describes how to use MapReduce to run a WordCount program in the Eclipse development plug-in.

Procedure

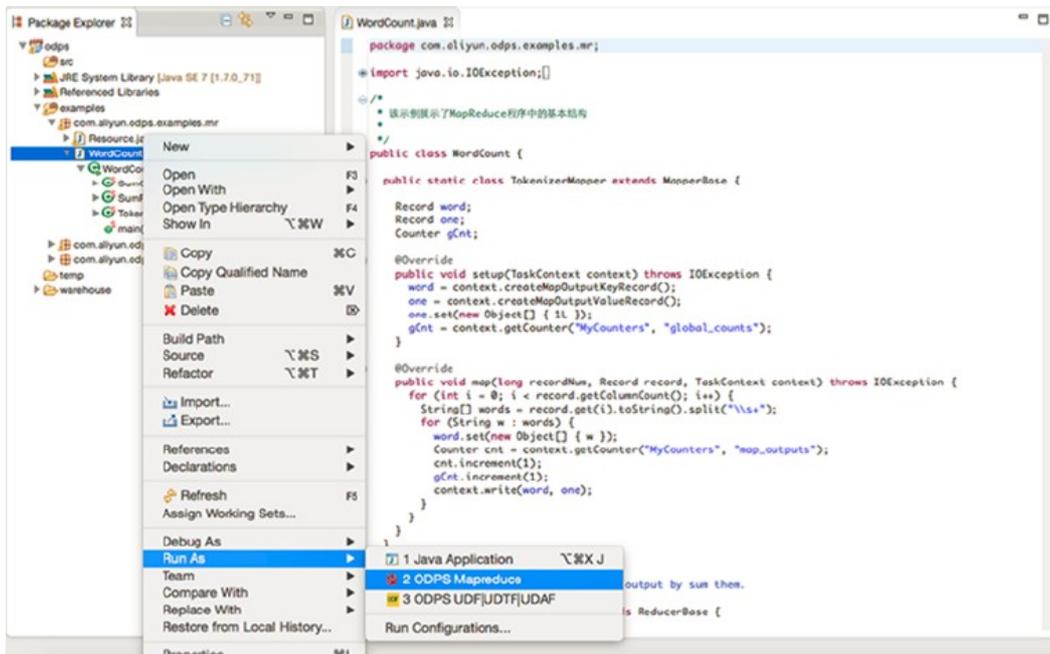
1. On the Package Explorer tab of the Eclipse development plug-in, choose **Examples > com.aliyun.odps.examples.mr > WordCount.java** from the navigation tree, as shown in the following figure.

Choose the WordCount program



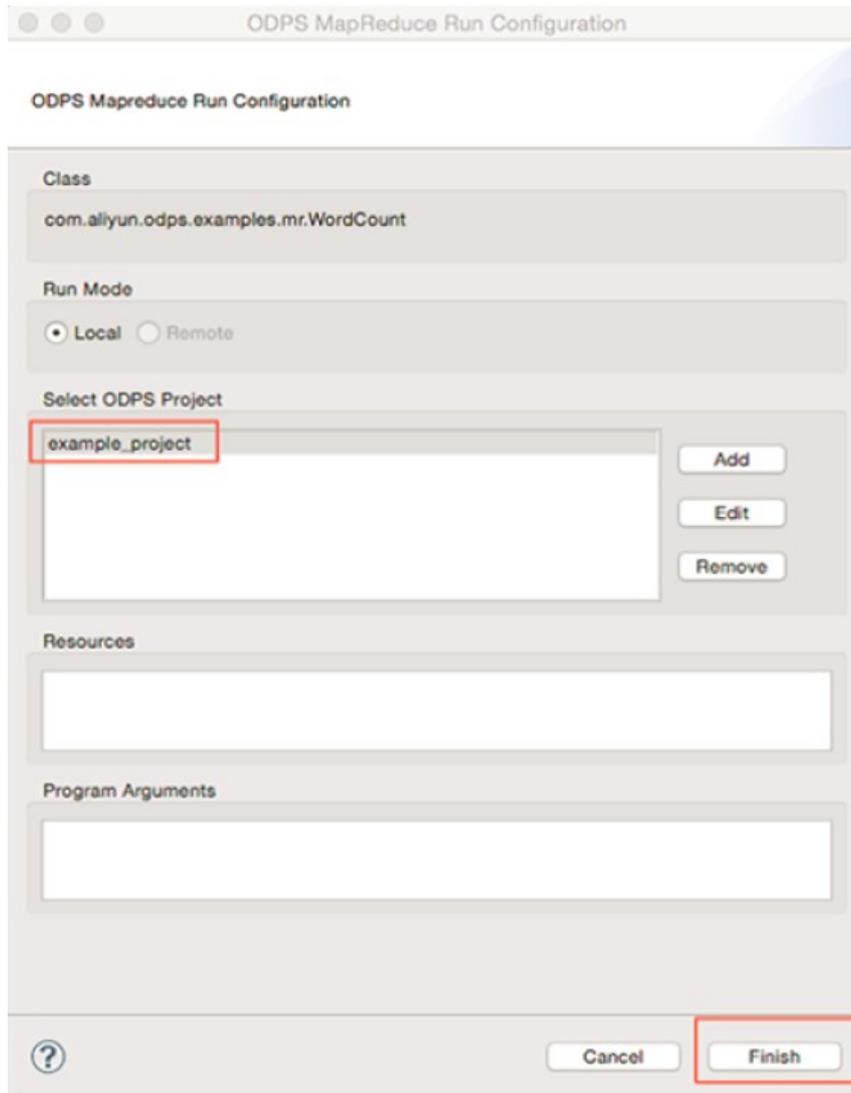
- 2. Right-click the WordCount.java program and choose Run As ODPs Mapreduce, as shown in the following figure.

Run the WordCount program (1)

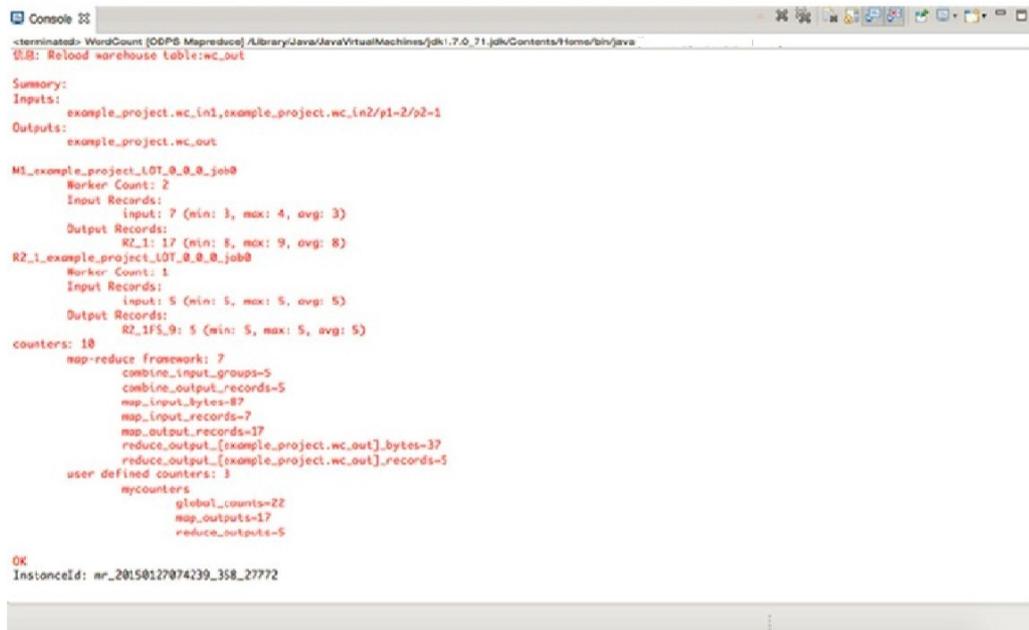


- 3. In the ODPs MapReduce Run Configuration dialog box, add example_project to the Select ODPs Project section and click Finish, as shown in the following figure.

Run the WordCount program (2)



4. After you run the WordCount program, the result is displayed, as shown in the following figure.
Execution result



```
<terminated> WordCount [ODPS MapReduce] /Library/Java/JavaVirtualMachines/jdk1.7.0_71.jdk/Contents/Home/bin/java
0.8: Reload warehouse table:wc_out

Summary:
Inputs:
  example_project.wc.in1,example_project.wc.in2/p1=2/p2=1
Outputs:
  example_project.wc.out

M1_example_project_LOT_0_0_0_job0
  Worker Count: 2
  Input Records:
    input: 7 (min: 3, max: 4, avg: 3)
  Output Records:
    R2_1: 17 (min: 8, max: 9, avg: 8)

R2_1_example_project_LOT_0_0_0_job0
  Worker Count: 1
  Input Records:
    input: 5 (min: 5, max: 5, avg: 5)
  Output Records:
    R2_IFS_9: 5 (min: 5, max: 5, avg: 5)

counters: 10
  map-reduce framework: 7
    combine_input_groups=5
    combine_output_records=5
    map_input_bytes=87
    map_input_records=7
    map_output_records=17
    reduce_output_[example_project.wc.out].bytes=37
    reduce_output_[example_project.wc.out].records=5
  user defined counters: 3
    mycounters
      global_counts=22
      map_outputs=17
      reduce_outputs=5

OK
InstanceId: mr_20150127074239_358_27772
```

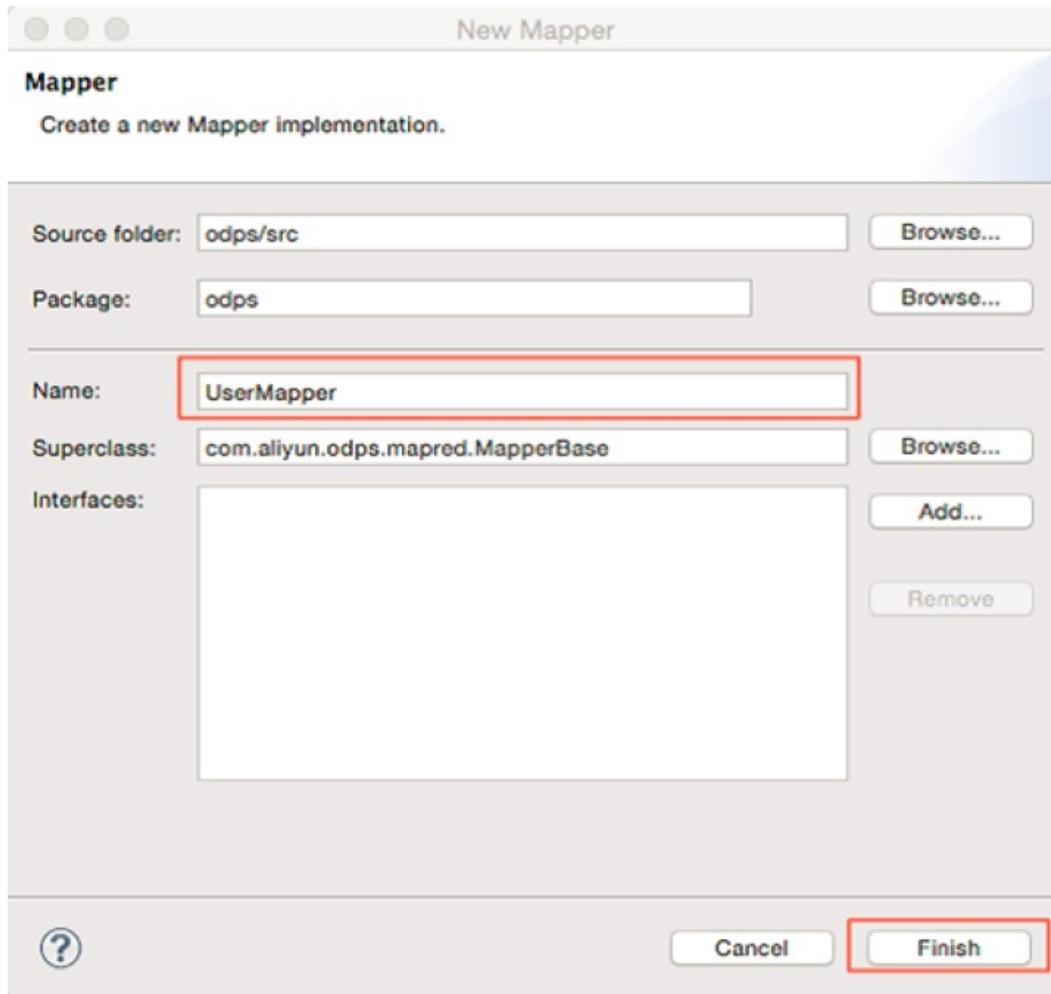
1.22.2.3.2. Run a custom MapReduce program

This topic provides an example on how to run a custom MapReduce program by using the Eclipse development plug-in.

Procedure

1. On the Package Explorer tab of the Eclipse development plug-in, choose `odps > src` from the navigation tree. Right-click `src` and choose **New > Mapper**.
2. In the New Mapper dialog box, enter the name of the Mapper class and click **Finish**, as shown in the following figure.

New Mapper dialog box

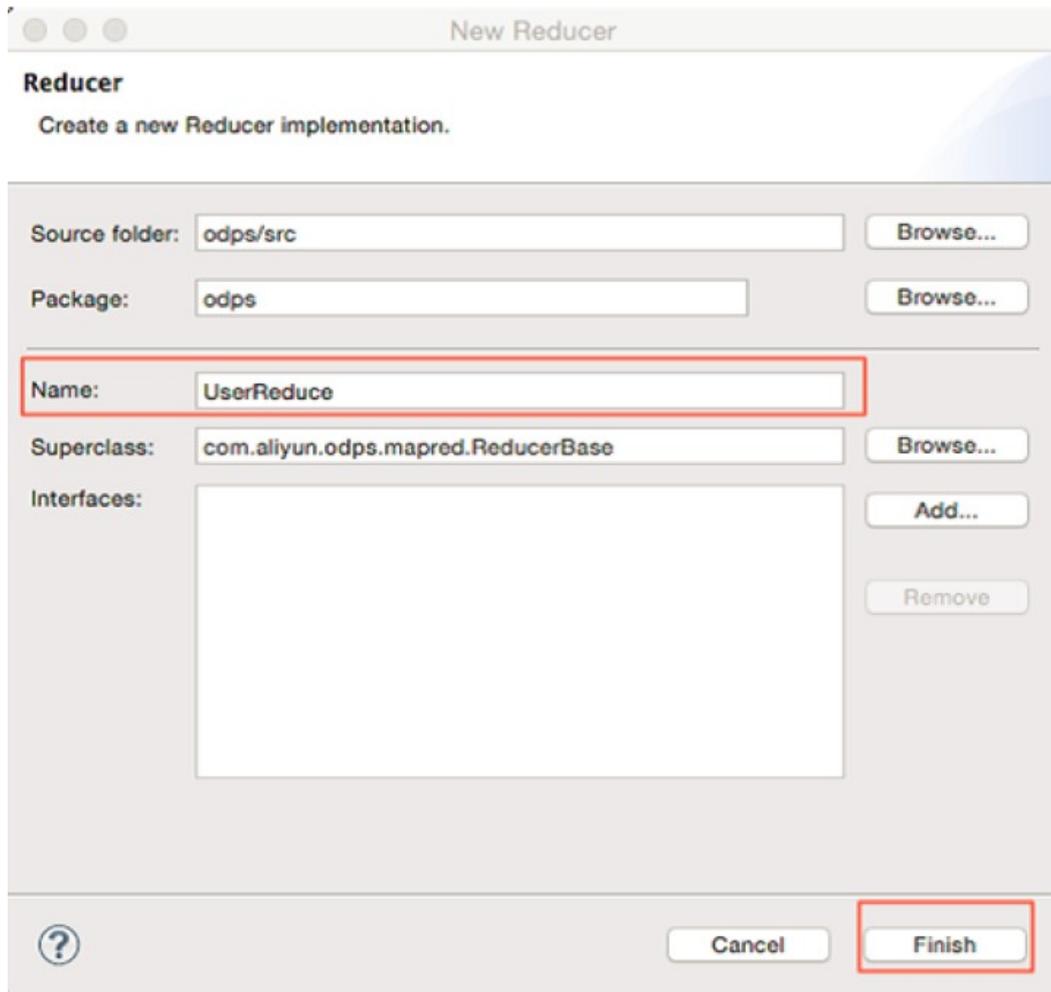


3. The UserMapper.java file is generated in the src folder shown in the navigation tree on the Package Explorer tab. The file is a Mapper class template. By default, the package name in the template is odps. The template contains the following information:

```
package odps;
import java.io.IOException;
import com.aliyun.odps.counter.Counter; import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.MapperBase;
public class UserMapper extends MapperBase {
Record word; Record one; Counter gCnt;
@Override
public void setup(TaskContext context) throws IOException {
word = context.createMapOutputKeyRecord(); one = context.createMapOutputValueRecord(); one.set(new Object[] { 1L });
gCnt = context.getCounter("MyCounters", "global_counts");
}
@Override
public void map(long recordNum, Record record, TaskContext context) throws IOException {
for (int i = 0; i < record.getColumnCount(); i++) { String[] words = record.get(i).toString().split("\\s+"); for (String w : words) {
word.set(new Object[] { w });
Counter cnt = context.getCounter("MyCounters", "map_outputs"); cnt.increment(1);
gCnt.increment(1); context.write(word, one);
}
}
}
@Override
public void cleanup(TaskContext context) throws IOException {
}
}
```

4. On the Package Explorer tab of the Eclipse development plug-in, choose odps > src from the navigation tree. Right-click src and choose **New > Reduce**.
5. In the New Reducer dialog box, enter the name of the Reducer class and click **Finish**, as shown in the following figure.

Reducer



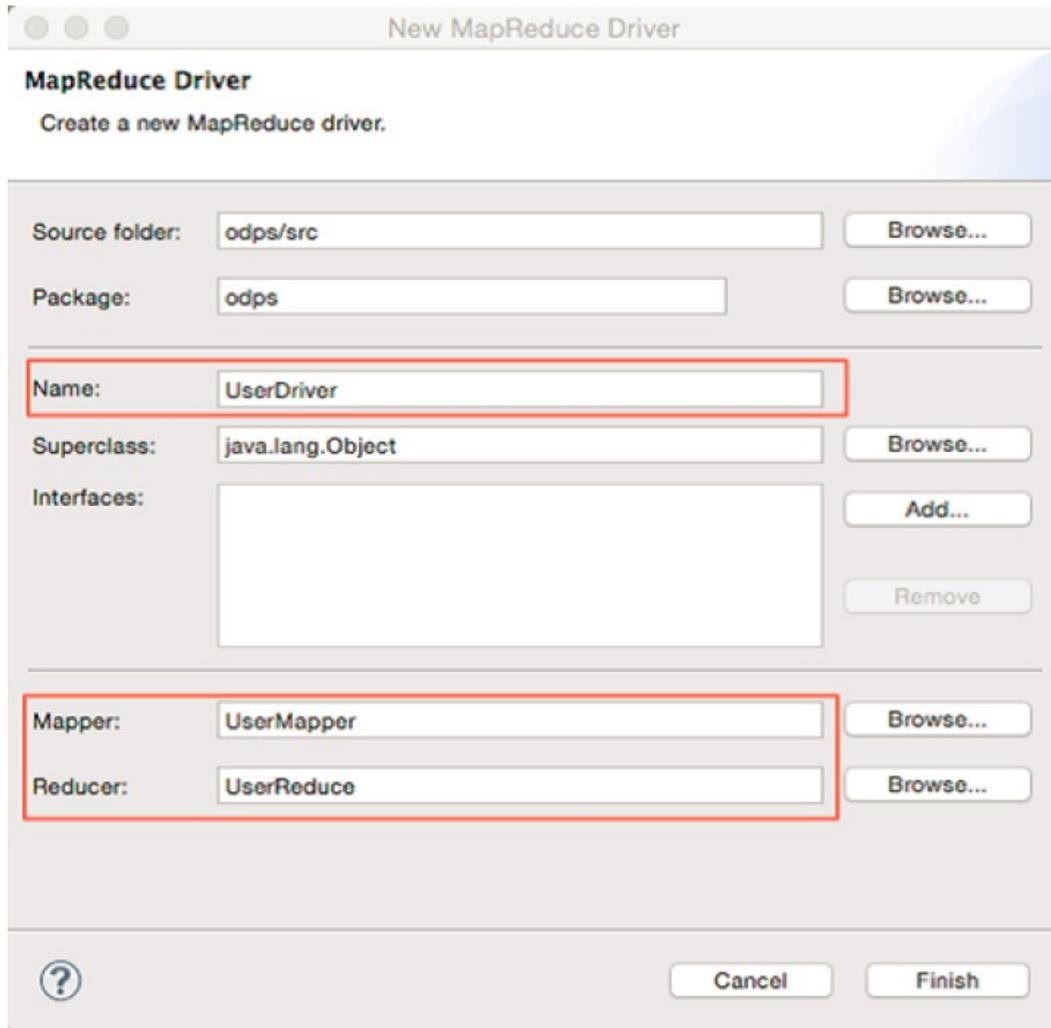
 **Note** In this example, the name of the Reducer class is UserReducer.

6. On the Package Explorer tab, the UserReducer.java file is generated in the src folder. The file is a Reducer class template. By default, the package name in the template is odps. The template contains the following information:

```
package odps;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.counter.Counter;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.ReducerBase;
public class UserReducer extends ReducerBase {
private Record result; Counter gCnt;
@Override
public void setup(TaskContext context) throws IOException { result = context.createOutputRecord();
gCnt = context.getCounter("MyCounters", "global_counts");
}
@Override
public void reduce(Record key, Iterator<Record> values, TaskContext context) throws IOException {
long count = 0;
while (values.hasNext()) { Record val = values.next(); count += (Long) val.get(0);
}
result.set(0, key.get(0)); result.set(1, count);
Counter cnt = context.getCounter("MyCounters", "reduce_outputs"); cnt.increment(1);
gCnt.increment(1);
context.write(result);
}
@Override
public void cleanup(TaskContext context) throws IOException {
}
}
```

7. On the Package Explorer tab of the Eclipse development plug-in, choose odps > src from the navigation tree. Right-click src and choose **New > MapReduce Driver**.
8. In the New MapReduce Driver dialog box, Specify Name, Mapper, and Reducer, and click **Finish**, as shown in the following figure.

MapReduce Driver

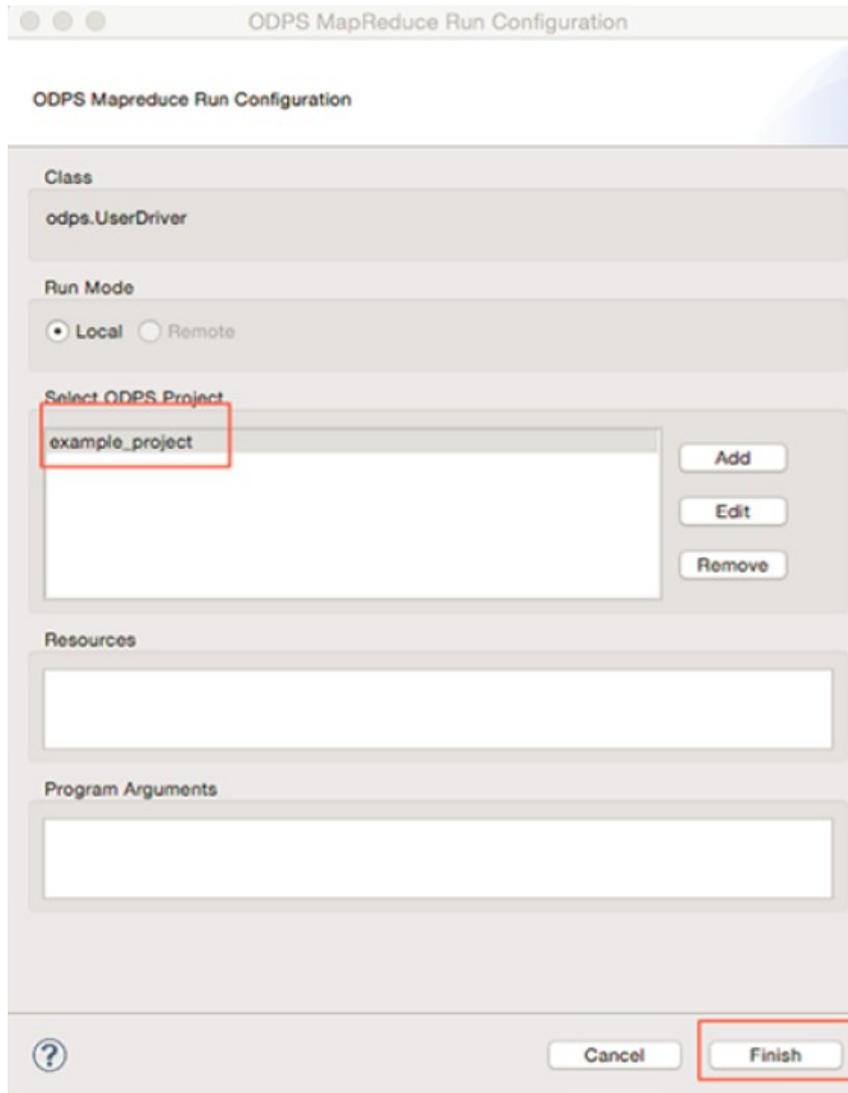


9. On the Package Explorer tab, the MyDriver.java file is generated in the src folder. The file is a MapReduce Driver class template. By default, the package name in the template is odps. The template contains the following information:

```
package odps;
import com.aliyun.odps.OdpsException;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.examples.mr.WordCount.SumCombiner;
import com.aliyun.odps.examples.mr.WordCount.SumReducer;
import com.aliyun.odps.examples.mr.WordCount.TokenizerMapper;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.RunningJob;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
public class UserDriver {
public static void main(String[] args) throws OdpsException { JobConf job = new JobConf();
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(SumCombiner.class);
job.setReducerClass(SumReducer.class);
job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
InputUtils.addTable(
TableInfo.builder().tableName("wc_in1").cols(new String[] { "col2", "col3" }).build(), job);
InputUtils.addTable(TableInfo.builder().tableName("wc_in2").partSpec("p1=2/p2=1").build(), job);
OutputUtils.addTable(TableInfo.builder().tableName("wc_out").build(), job);
RunningJob rj = JobClient.runJob(job); rj.waitForCompletion();
}
}
```

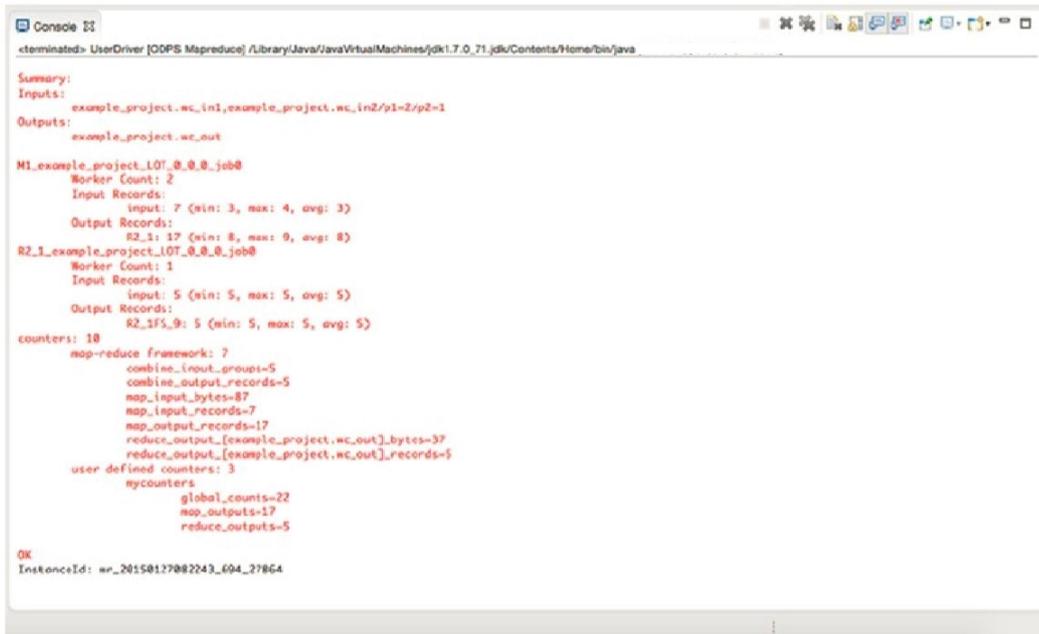
10. Run the MapReduce program. Right-click the UserDriver.java file, choose **Run As > ODPS MapReduce**, and then click **OK**. The ODPS MapReduce Run Configuration dialog box is displayed, as shown in the following figure.

ODPS MapReduce Run Configuration



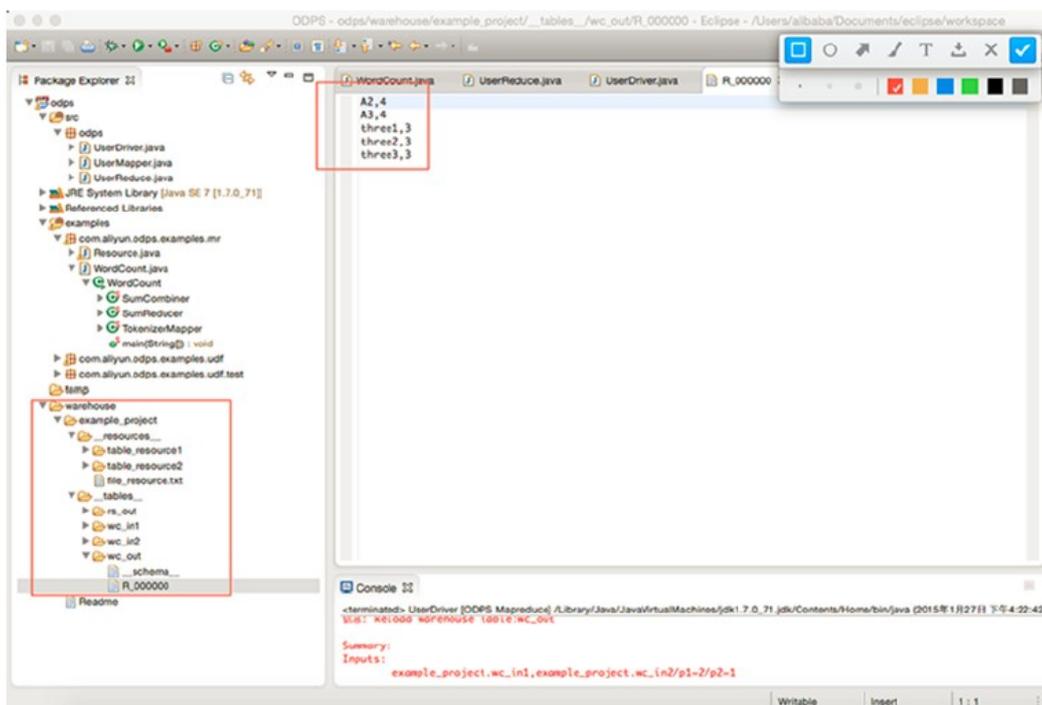
11. Add example_project in Select ODPS Project and click **Finish** to run the MapReduce program in local mode. If the following information is displayed, the MapReduce program properly runs in local mode.

Console



- Right-click the *warehouse* folder in the navigation tree on the Package Explorer tab and select Refresh to view the output result, as shown in the following figure.

Output result

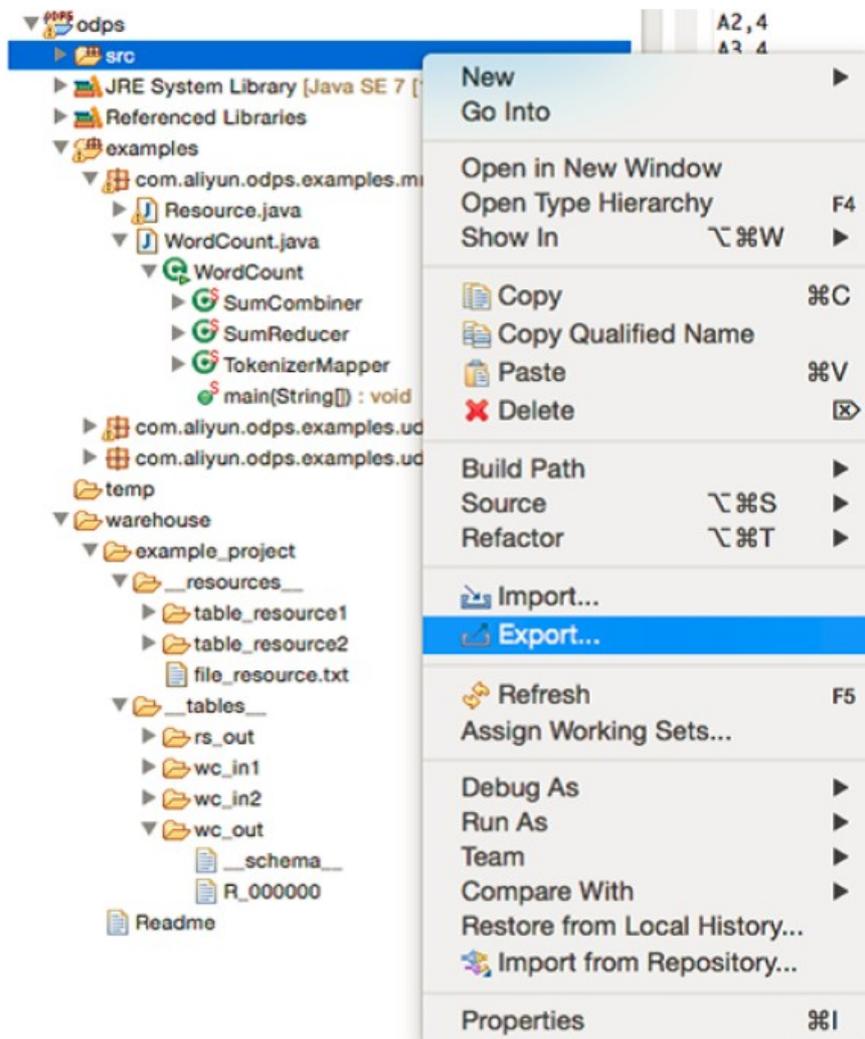


Note wc_out is the output folder, and R_000000 is the result file. After you perform local debugging to verify that the output result is correct, you can use the export feature of the Eclipse development plug-in to package the MapReduce program for future use in a distributed environment.

- Export the MapReduce program package.

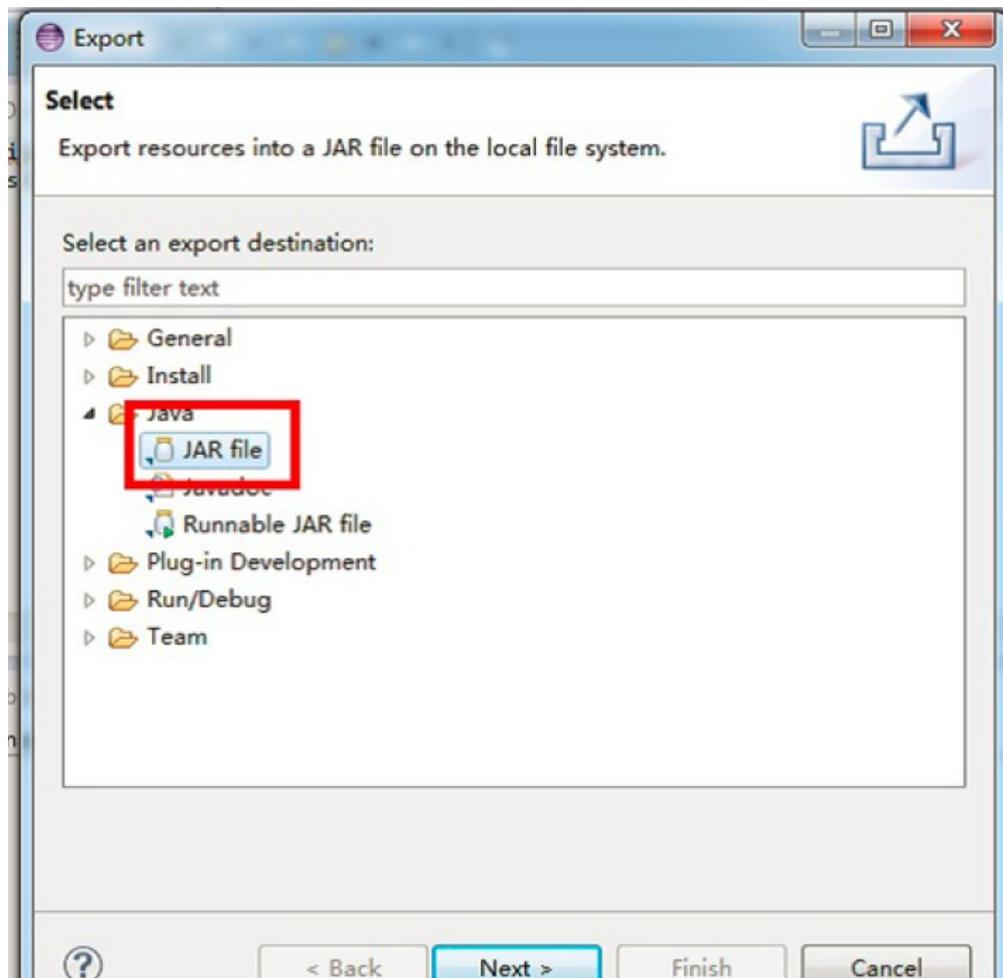
- i. On the Package Explorer tab of the Eclipse development plug-in, choose odps > src from the navigation tree. Right-click src and select **Export**, as shown in the following figure.

Export dialog box



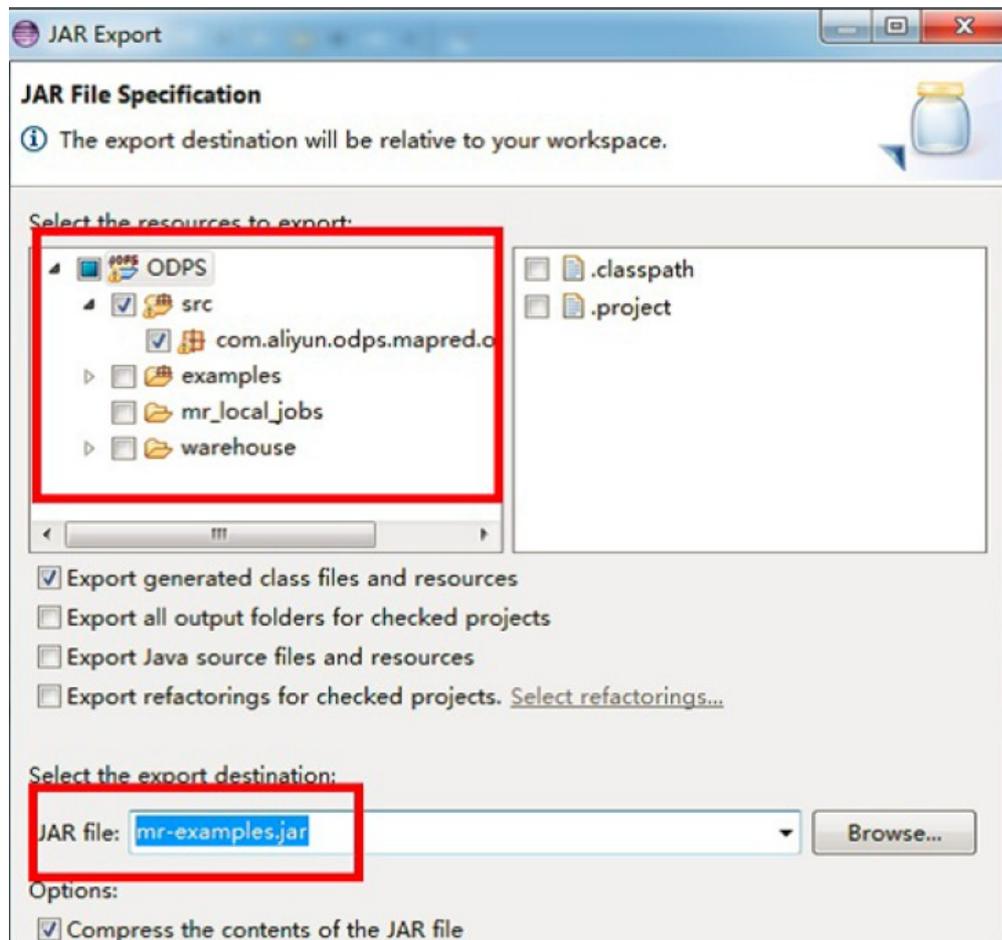
- ii. In the Export dialog box, choose Java > JAR file in Select an export destination and click Next, as shown in the following figure.

JAR Export dialog box



- iii. In the JAR Export dialog box, choose ODPS > src > com.aliyun.odps.mapred.open.example, and specify the mr-examples.jar file as the MapReduce program package, as shown in the following figure.

Select the source and destination packages



Note In this example, the name of the MapReduce program package is mr-examples.jar. You can name the package based on your business requirements.

- iv. Click Finish. The export process is complete.
14. If you want to create a project, you can create a folder at the same directory level as the example_project folder in the warehouse folder. The following figure shows the directory structure.

```
<warehouse>
└─ example_project
  └─ <_tables_>
    │ └─ table_name1
    │   └─ data
    │     │
    │     └─ <_schema_>
    │       │
    │       └─ table_name2
    │         └─ partition_name=partition_value
    │           └─ data
    │             │
    │             └─ <_schema_>
    │               │
    └─ <_resources_>
      │
      └─ table_resource_name
        │ └─ <_ref_>
        │
        └─ file_resource_name
```

Examples of information in schema files:

```
project=project_name table=table_name
columns=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING
-- Information in the schema file of a non-partitioned table:
project=project_name table=table_name
columns=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING partitions=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING
-- Information in the schema file of a partitioned table:
-- The following data types are supported: BIGINT, DOUBLE, BOOLEAN, DATETIME, and STRING. These data types correspond to LONG, DOUBLE, BOOLEAN, Java.Util.Date, and Java.Lang.STRING in Java.
```

Example of a data file:

```
1,1.1,true,2015-06-04 11:22:42 896,hello world
\n,\n,\n,\n,\n
-- The time is accurate to milliseconds. \N represents NULL for all data types.
```

Note

- Before the MapReduce program runs in local mode, the program automatically checks for data tables or resources in the warehouse folder first. If the tables or resources are not found, the program downloads the tables or resources from a server to the *warehouse* folder.
- After you run the MapReduce program, right-click the *warehouse* folder in the navigation tree on the Package Explorer tab and select Refresh to view the output result.

1.22.2.4. UDF development and running example

1.22.2.4.1. Local debug UDF programs

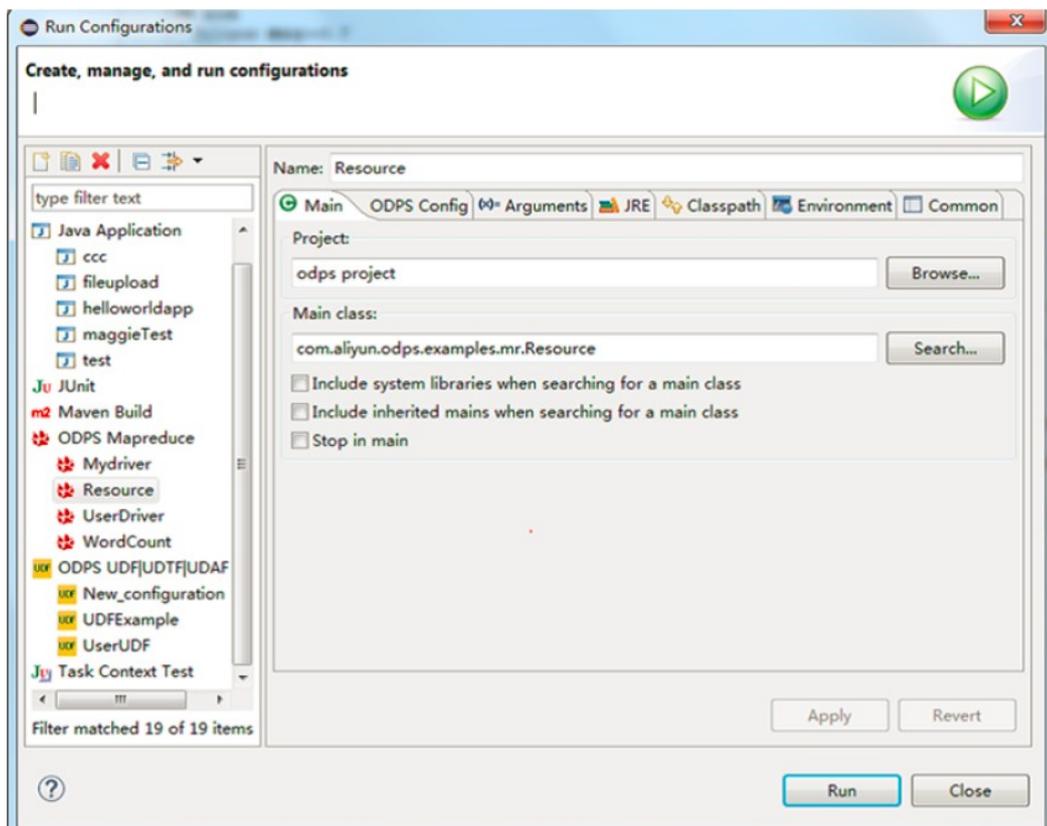
1.22.2.4.1.1. Run a UDF from the menu bar

This topic describes how to run a user-defined function (UDF) from the menu bar of the Eclipse development plug-in.

Procedure

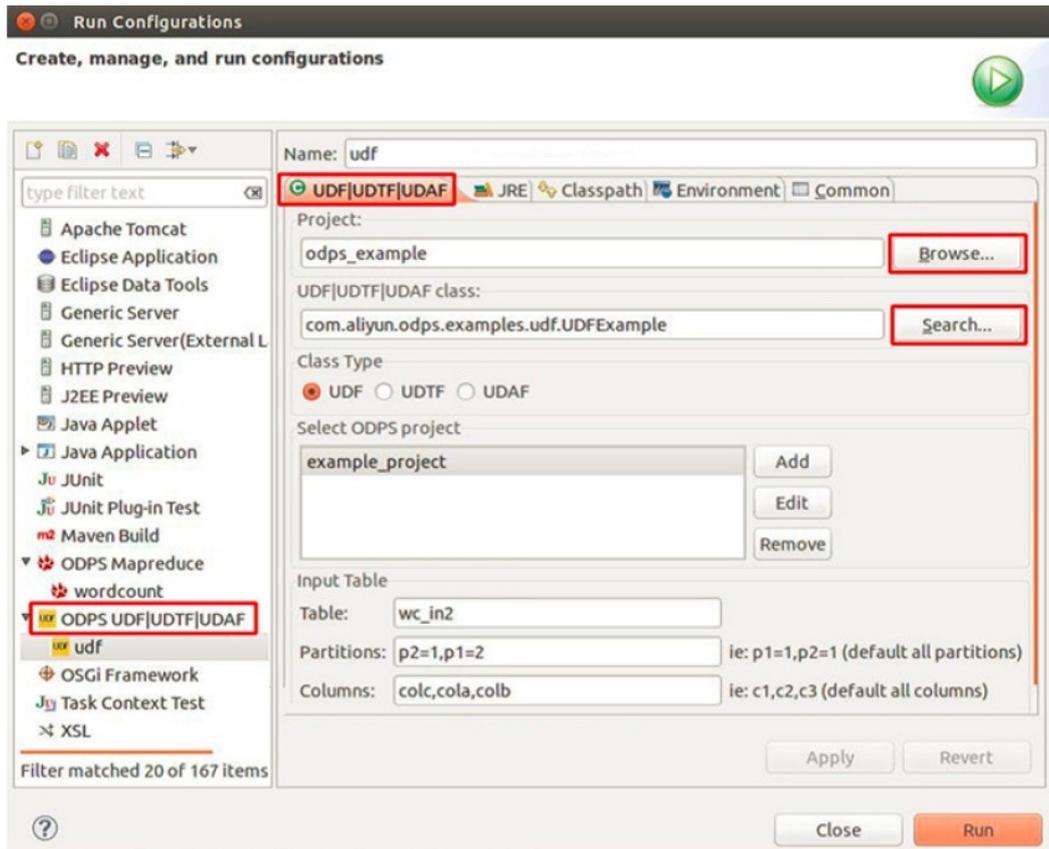
1. Choose **Run > Run Configurations** from the menu bar. The Run Configurations dialog box appears, as shown in the following figure.

Run Configurations



2. In the left-side navigation tree, right-click ODPs UDF|UDTF|UDAF and select New. On the ODPs UDF|UDTF|UDAF tab, select the project, UDF class, class type, and ODPs project, and enter the table information, as shown in the following figure.

Run Configurations 2



Note In the preceding figure, the Table parameter indicates the input table of the UDF. The Partitions parameter indicates the partitions from which you want to read data. Partition names are separated by commas (.). The Columns parameter indicates the columns that are passed as parameters of the UDF. Column names are separated by commas (.).

3. Click **Run**. The running result is displayed in the console, as shown in the following figure.

Console



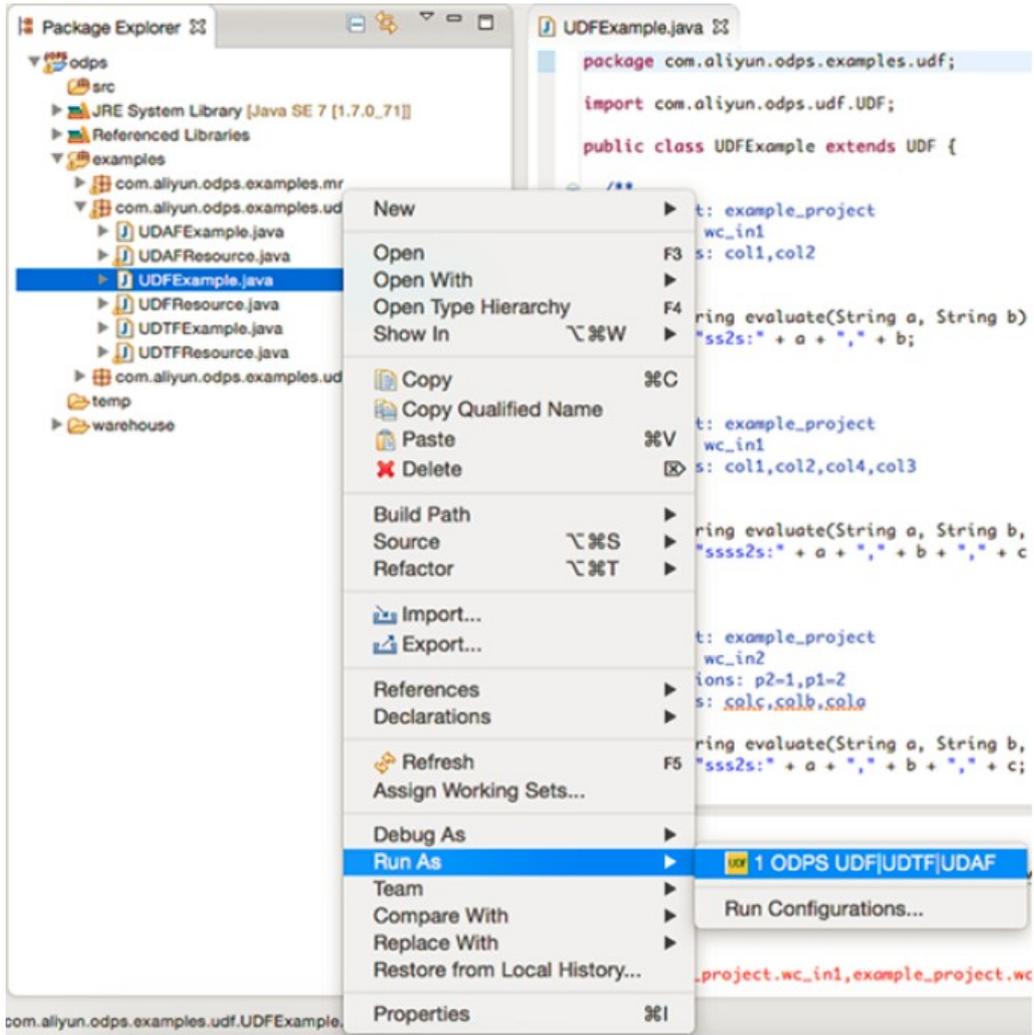
1.22.2.4.1.2. Use the right-click shortcut menu to run a UDF

This topic describes how to use the right-click shortcut menu to run a user-defined function (UDF) in the Eclipse development plug-in.

Procedure

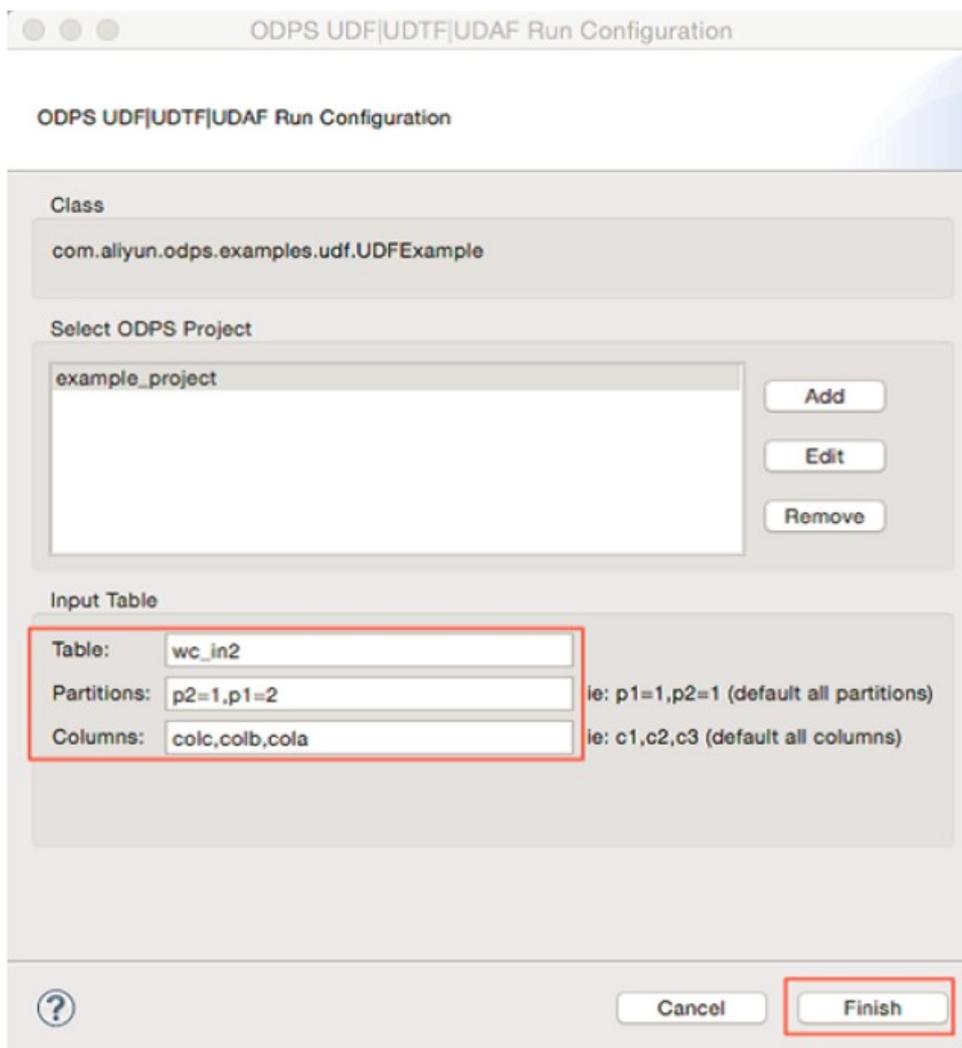
1. On the Package Explorer tab of the Eclipse development plug-in, right-click a Java file in the navigation tree, such as UDFExample.java, and choose **Run As > Run UDF|UDAF|UDTF**, as shown in the following figure.

Short cut menu



2. In the ODPS UDF|UDTF|UDAF Run Configuration dialog box, configure related parameters, as shown in the following figure.

ODPS UDF|UDTF|UDAF Run Configuration dialog box



Note In the ODPS UDF|UDTF|UDAF Run Configuration dialog box, the Table parameter indicates the input table of the UDF. The Partitions parameter indicates the partitions from which you want to read data. Separate multiple partitions with commas (.). The Columns parameter indicates the columns that are passed as the parameters of the UDF. Separate multiple columns with commas (.).

3. Click **Finish** to run the UDF and obtain the output result.

1.22.2.4.2. Run a UDF

This topic describes how to run a user-defined function (UDF) in the Eclipse development plug-in.

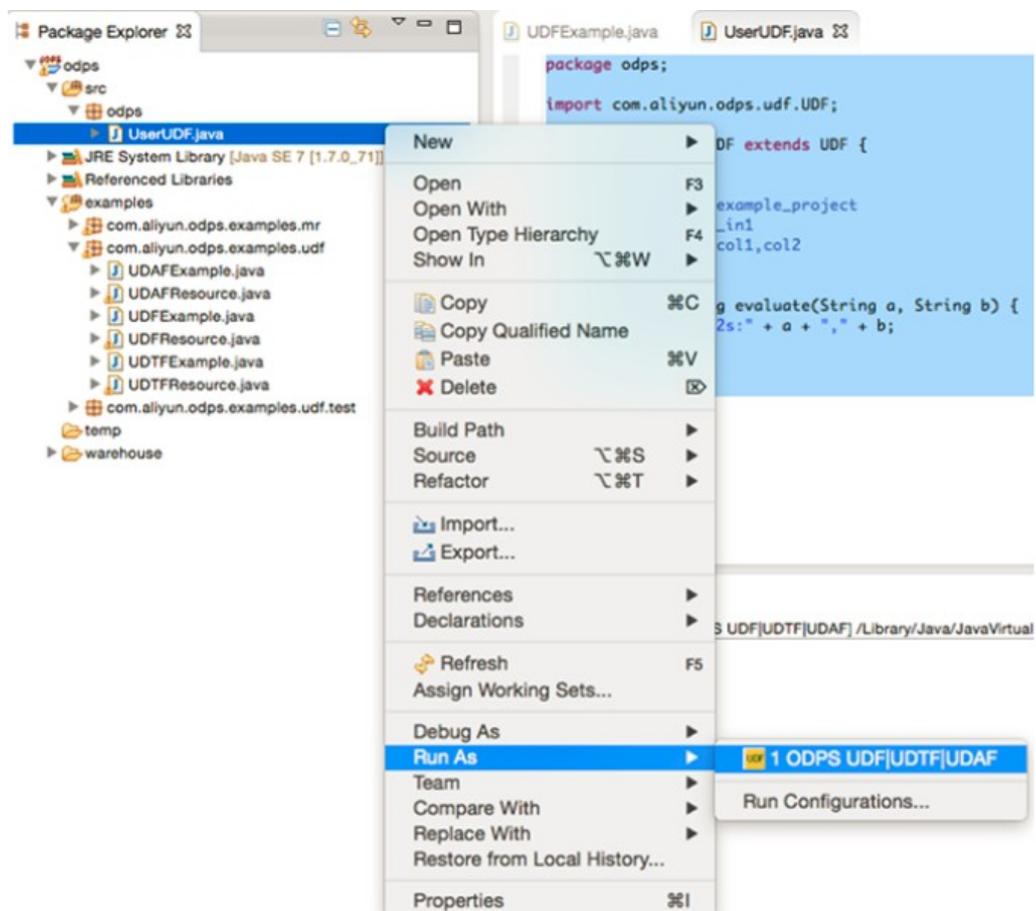
Procedure

1. Right-click a project and choose **New > UDF**. You can also choose **File > New > UDF** from the menu bar. Then, enter a UDF class name and click **Finish**. A Java file with the same name as the UDF class is generated in the `src` directory. Edit the content of the Java file. Sample content in the Java file:

```
package odps;
import com.aliyun.odps.udf.UDF;
public class UserUDF extends UDF {
/**
 * project: example_project
 * table: wc_in1
 * columns: col1,col2
 *
 */
public String evaluate(String a, String b) { return "ss2s:" + a + "," + b;
}
}
```

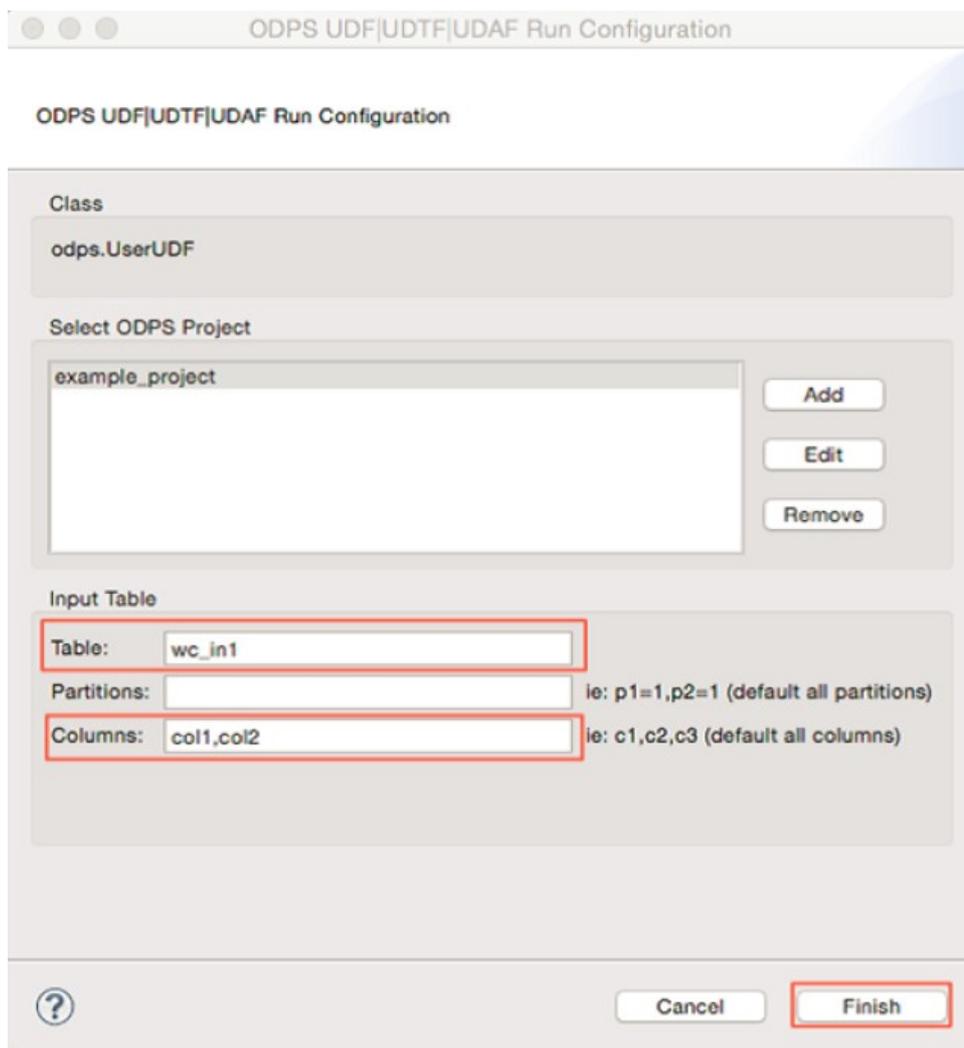
2. Right-click the Java file, such as the UserUDF.java file, and choose **Run As > ODPS UDF|UDTF|UDAF**, as shown in the following figure.

Run a UDF (1)



3. In the dialog box that appears, configure the parameters shown in the following figure.

Run a UDF (2)



4. Click **Finish** to obtain the result.

```
ss2s:A1,A2  
ss2s:A1,A2  
ss2s:A1,A2  
ss2s:A1,A2
```

Note This example shows how to run a UDF. You can use the same method to run a user-defined table-valued function (UDTF).

1.22.2.5. Graph operation example

After you create a MaxCompute project, you can compile your own Graph program and perform the following operations to complete local debugging: This topic provides an example about how to run a Graph.

Context

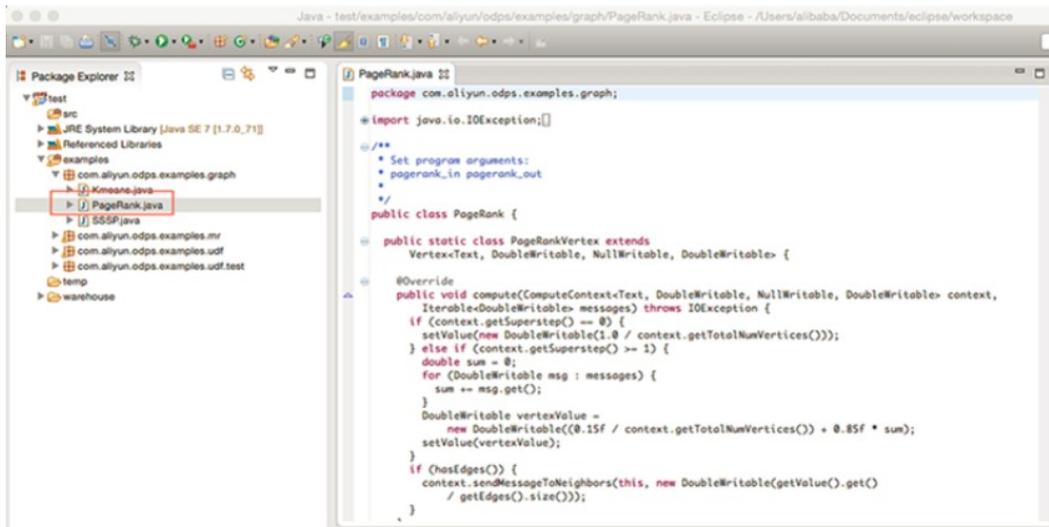
In this example, you can use the PageRank.java file provided by a plug-in to perform local debugging.

Procedure

1. On the Package Explorer tab, choose example > com.aliyun.odps.examples.graph > PageRank.java, as

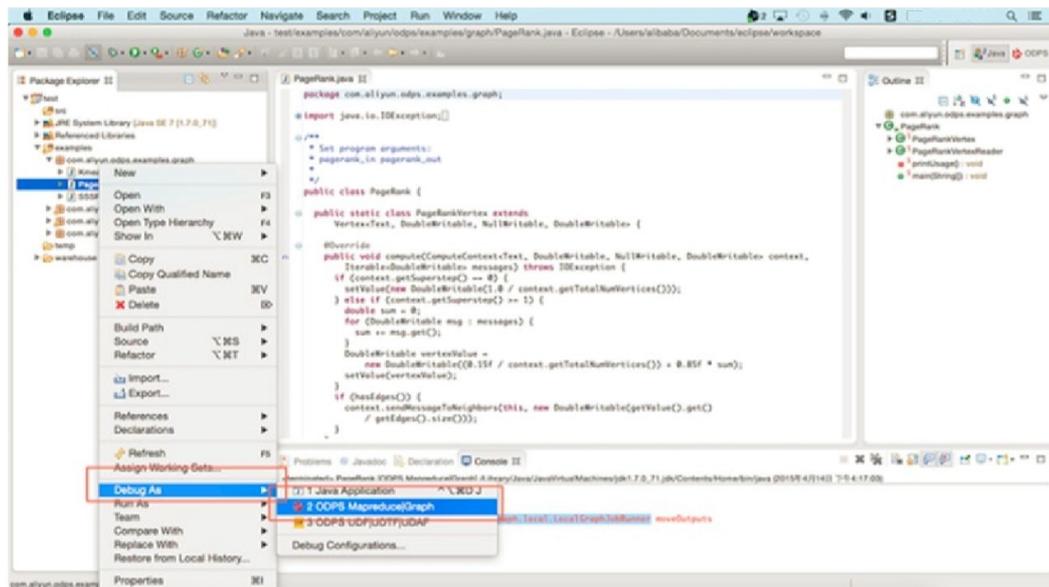
shown in the following figure.

Choose PageRank.java



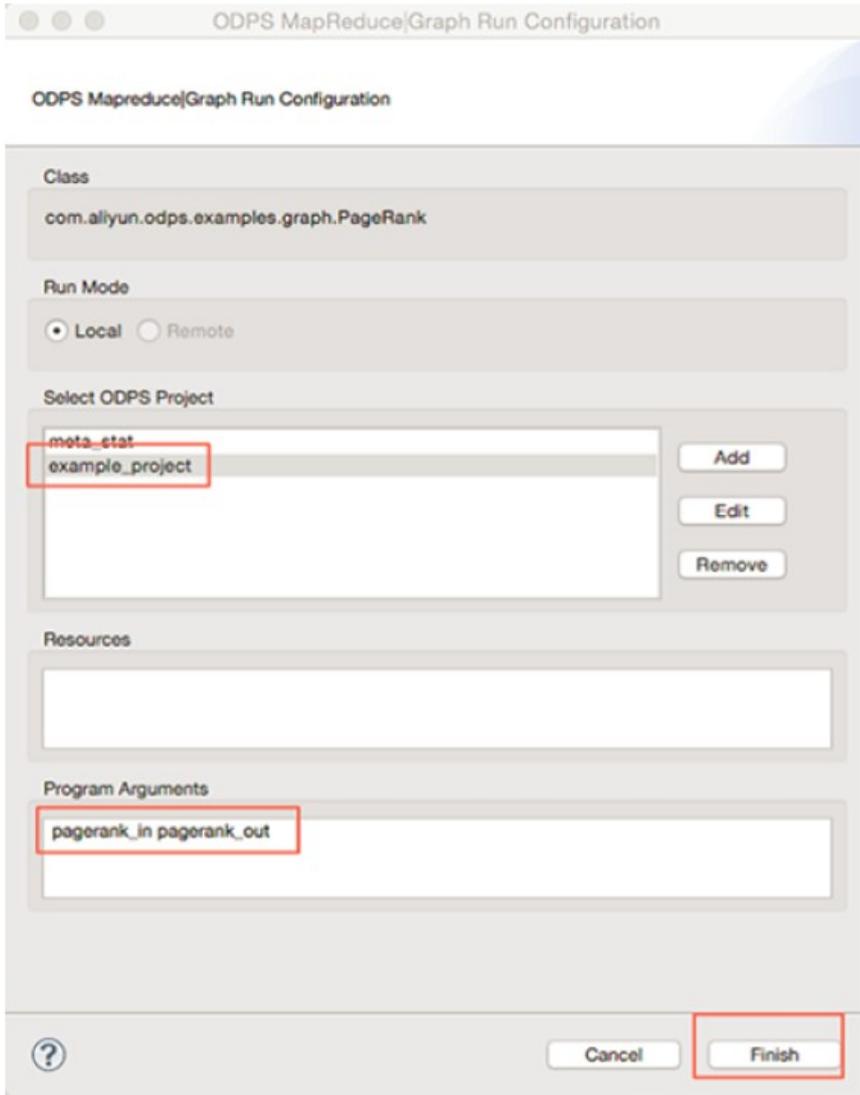
2. Right-click the PageRank.java file and choose **Debug As > ODPS MapReduce|Graph**, as shown in the following figure.

Choose Debug As > ODPS MapReduce|Graph



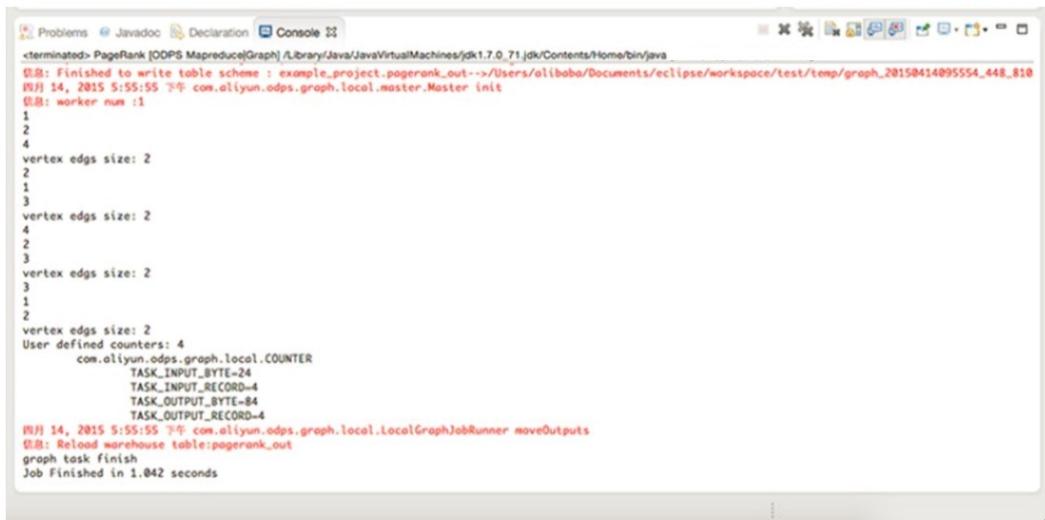
3. In the ODPS MapReduce/Graph Run Configuration dialog box, set the parameters to the values shown in the following figure.

ODPS MapReduce/Graph Run Configuration dialog box



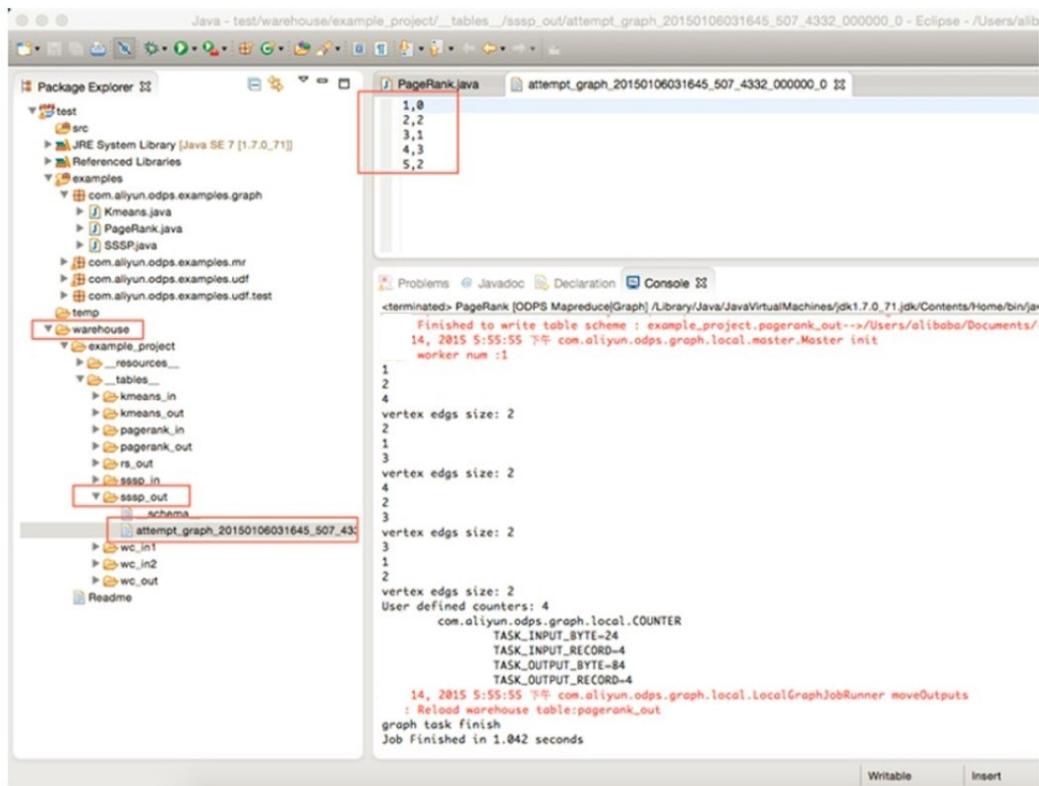
4. Click **Finish** to view the job running result, as shown in the following figure.

Running result



View the local computing result, as shown in the following figure.

Local computing result



After local debugging succeeds, you can package the program, upload the package to MaxCompute as a JAR resource, and submit a Graph job.

Note

- For more information about the packaging process, see [MapReduce running example](#).
- For more information about the directory structure of local results, see [MapReduce running example](#).
- For more information about how to upload JAR resources, see [Compile and run a Graph job](#).

1.23. MaxCompute feature enhancement packages

1.23.1. Content moderation

This topic describes the content moderation enhancement package.

Purpose

The content moderation enhancement package provides an end-to-end general-purpose or industrial solution for content moderation in big data fields. This package is developed for Apsara Stack customers who have big data and content moderation requirements. In this solution, you can use MaxCompute to capture data, including structured data and unstructured data. You can also identify and review the content related to politics, pornography, and violence based on the content moderation engine.

Description

- Content moderation

Content Moderation is a service that intelligently moderates the content of multimedia objects, such as images, videos, text, and audios in various scenarios. This service helps you effectively reduce the risk of content violations.

Common detection scenarios include intelligent pornography detection, terrorism recognition, graphic advertising recognition, logo recognition, facial recognition, QR code recognition, OCR image-text recognition, text-based anti-spam, voice-based anti-spam, and file content-based anti-spam.

Content Moderation provides the website detection feature. This feature automatically detects risky and illicit content on your website on a regular basis. The OSS violation detection feature is provided to perform pornography- and violence-related content detection on images and videos in your specified Object Storage Service (OSS) buckets. You can also directly call the Content Moderation API to submit machine identification tasks for specific scenarios.

- Types of content moderation

Content moderation is classified into the following types based on the data types and API definitions:

- Image moderation.
- Video moderation.
- Text moderation.
- Audio moderation.
- File moderation. After you submit a file moderation task, file content is automatically parsed to moderate images and text in the file.

- Unstructured data processing in MaxCompute

In addition to structured data, MaxCompute also allows you to process unstructured data. You can use external tables to process unstructured data stored in data storage services, such as OSS and Tablestore. You can also use volumes to store and process unstructured data in MaxCompute.

In content moderation scenarios, you can directly use Content Moderation to process unstructured data in OSS. In volume or table scenarios, we recommend that you perform content moderation on data in MaxCompute.

Examples

Raw data:

```
localtest_byok>read table_iacs_demo;
```

The following result is returned:

```
+-----+
| data  |
+-----+
| aaaa  |
| Obama and Trump met at White House yesterday. |
| Hi, a booty call? |
| You are such a rubbish, really heartless. |
| If you have free time and want to find a part-time job, you can call me. |
| bbbb  |
+-----+
```

The following content is provided after content moderation is performed:

```
localtest_byok>select IAcS_demo(data, "demo") as (code,msg,content,filteredContent,suggestion,label,results) from table_iaCS_demo;
```

The following result is returned:

```
+-----+-----+-----+-----+-----+-----+
| code | msg | content | filteredcontent | suggestion | label | results |
+-----+-----+-----+-----+-----+-----+
| 200 | OK | aaaa | NULL | pass | normal | [{"rate":99.91,"suggestion":"pass","label":"normal","scene":"antis
pam"}] |
| 200 | OK | Obama and Trump met at White House yesterday. | Obama and *** met at White House yesterday. | bloc
k | customized | [{"rate":99.91,"suggestion":"block","details":{"contexts":{"libCode":"4386012","libName":"de
mo"},"context":"Trump"},"label":"customized"},"label":"customized","scene":"antis
pam"}] |
| 200 | OK | Hi, a booty call? | NULL | block | porn | [{"rate":99.91,"suggestion":"block","details":{"contexts":{"
context":"Hi, a booty call?"},"label":"porn"},"label":"porn","scene":"antis
pam"}] |
| 200 | OK | You are such a rubbish, really heartless. | NULL | block | abuse | [{"rate":89.1,"suggestion":"block","
details":{"label":"abuse"},"label":"abuse","scene":"antis
pam"}] |
| 200 | OK | If you have free time and want to find a part-time job, you can call me. | NULL | pass | normal | [{"ra
te":99.91,"suggestion":"pass","label":"normal","scene":"antis
pam"}] |
| 200 | OK | bbbb | NULL | pass | normal | [{"rate":99.91,"suggestion":"pass","label":"normal","scene":"anti
spam"}] |
+-----+-----+-----+-----+-----+-----+
```

1.23.2. MCQA

1.23.2.1. Overview

This topic describes the architecture, key features, scenarios, and limits of MaxCompute Query Acceleration (MCQA).

Description

MCQA of MaxCompute accelerates the execution of small- and medium-sized query jobs and reduces the execution time from minutes to seconds. MCQA is compatible with other query features of MaxCompute.

MCQA allows you to perform ad hoc queries or business intelligence (BI) analysis by connecting mainstream BI tools or SQL clients to MaxCompute projects.

MCQA uses an independent resource pool that does not use quota groups. MCQA automatically identifies query jobs and shortens the job queue length, which improves user experience.

Key features

MCQA provides the following key features:

- Low-latency resource scheduling
 - Uses an efficient and low-latency resource scheduling policy and an independent resource pool.
- Automatic identification
 - Automatically identifies the size of query jobs. MaxCompute can use MCQA to accelerate the queries or process multiple query jobs at the same time and then return the query results. This allows you to analyze query jobs of different sizes or complexities.
- Syntax compatibility
 - Uses the same SQL syntax as MaxCompute.

- Selection of query methods

Allows the MaxCompute client to use MCQA or the offline mode to run query jobs. You can also configure to forcibly use MCQA in latency-sensitive scenarios.

Scenarios

The following table describes the scenarios for which MCQA is suitable.

Scenario	Description	Applicable scope
Ad hoc query	You can use MCQA to optimize the query performance of small- and medium-sized datasets (less than 100 GB) and perform low-latency queries on MaxCompute tables. This helps develop and analyze data.	You can specify query criteria based on your requirements, obtain query results, and adjust the query logic. In this scenario, the query latency must be within dozens of seconds. Users are data developers or data analysts who have mastered SQL skills and prefer to use the client tools that they are familiar with to analyze queries.
BI analysis	If you use MaxCompute to build an enterprise-class data warehouse, MaxCompute performs extract, transform, load (ETL) operations to process data into business-oriented and consumable aggregate data. MCQA features low latency and supports elastic concurrency and data caching. You can use MCQA with partitions and buckets in MaxCompute tables to run concurrent jobs, generate reports, analyze statistics, and analyze fixed reports at a low cost.	In most cases, the query object is aggregate data. This scenario is suitable for multidimensional queries, fixed queries, or high-frequency queries that contain small amounts of data. In this scenario, queries are latency-sensitive, and the results are returned in seconds. For example, the latency for most queries is less than 5 seconds. The time elapsed for each query varies based on the data size and query complexity.
Detailed queries and analysis of large amounts of data	MCQA automatically identifies the size of query jobs. MCQA can respond to and process small-sized jobs in a timely manner, and can allocate the resources required for large-sized jobs. This helps analysts run query jobs of different sizes and complexities.	In this scenario, large amounts of historical data is queried. The size of valid data that is queried is small, and the requirement for latency is low. Users are business analysts who need to gain business insights from data, explore potential business opportunities, and validate business assumptions.

Limits

MCQA supports only SELECT statements and does not support user-defined functions (UDFs). If you submit a statement or feature that MCQA does not support from the MaxCompute client, the MaxCompute client automatically rolls back to the common offline mode to execute the statement or feature. Other tools cannot roll back to the common offline mode to execute the submitted statement or feature that MCQA does not support.

The following table describes the limits of MCQA.

Item	Limit
Query	<ul style="list-style-type: none"> If more than 1,000 MCQA query jobs are executed at the same time, the system automatically rolls back these jobs to SQL query jobs. If you submit an MCQA query job from the MaxCompute client, the default timeout period is 30 seconds. If you submit an MCQA query job from the ad hoc query module of DataWorks, the default timeout period is 20 seconds. If the MCQA query job times out, the system automatically rolls back the MCQA query job to an SQL query job. MCQA can cache only data that is stored in ALIORC tables into memory to accelerate queries. You cannot use MCQA to query data from external tables.
Query concurrency	You can run a maximum of 120 concurrent MCQA query jobs in each MaxCompute project.

1.23.2.2. Usage notes

MaxCompute Query Acceleration (MCQA) is a built-in feature of MaxCompute. MCQA uses the native MaxCompute SQL syntax and supports MaxCompute built-in functions and permission systems. This topic describes how to use MCQA.

Context

You can use one of the following methods to enable the MCQA feature:

- Use the MaxCompute client. For more information, see [Enable MCQA on the MaxCompute client](#).
- Use the ad hoc queries or data analytics feature of DataWorks. By default, the MCQA feature is enabled for ad hoc queries or data analytics of DataWorks. For more information, see [Enable MCQA on the Ad-Hoc Query or Manually Triggered Workflow page](#).
- Use the MaxCompute Java Database Connectivity (JDBC) driver. For more information, see [JDBC](#).
- Use a MaxCompute SDK. This method requires dependencies in a pom.xml file. For more information, see [Enable MCQA by using MaxCompute SDK for Java](#).

Enable MCQA on the MaxCompute client

To enable the MCQA feature on the MaxCompute client, perform the following steps:

- Download the [client](#) package of the latest version.
- Decompress the package to a folder that contains the following subfolders:

```
bin/
conf/
lib/
plugins/
```

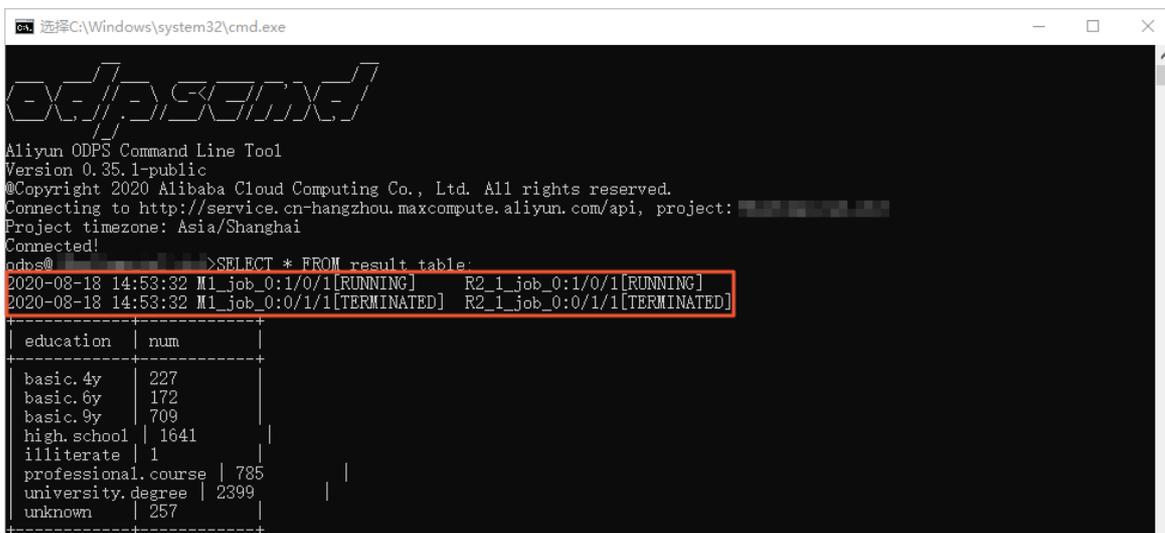
- Install and configure the client. For more information, see [Configure the client](#).
- Append the following command line to the odps_config.ini file in the conf folder.

```
enable_interactive_mode=true
```

- Run the odpscmd file in the bin directory in the Linux operating system or run the odpscmd.bat file in the bin directory in the Windows operating system. If the following information appears, the MaxCompute client runs properly.



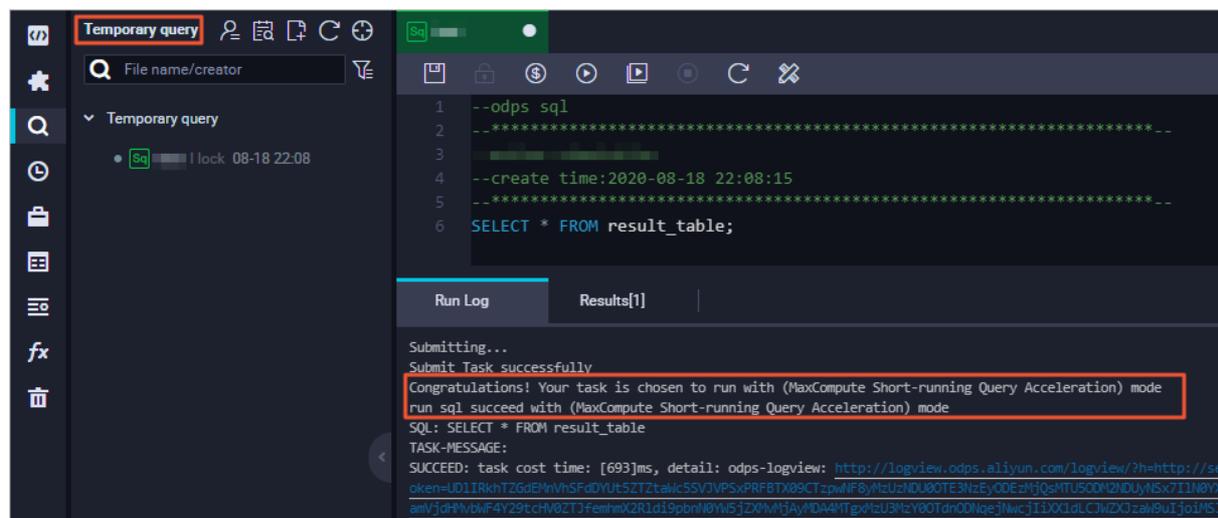
- Run a query job to verify that the MCQA feature is enabled. If the returned results contain the following information after you run the query job, the MCQA feature is enabled.



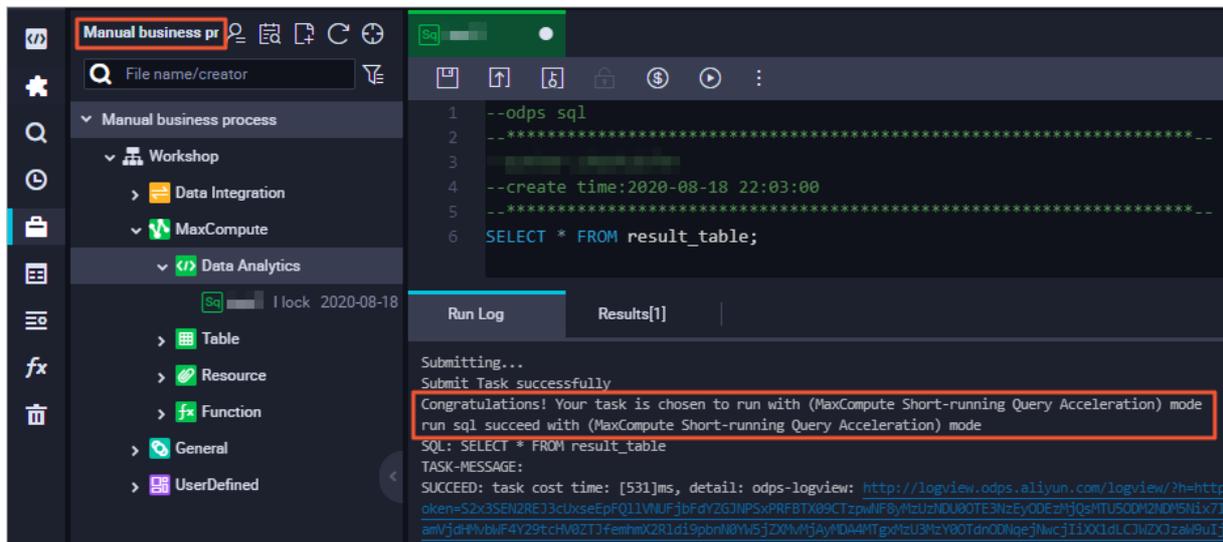
Enable MCQA on the Ad-Hoc Query or Manually Triggered Workflow page

By default, the MCQA feature is enabled on the Ad-Hoc Query or Manually Triggered Workflow page of DataWorks. Manual operations are not required. If you want to disable the MCQA feature, [submit a ticket](#).

Run a query job on the Ad-Hoc Query page. If the returned results contain the following information, the MCQA feature is enabled.



Run a query job on the **Manually Triggered Workflow** page. If the returned results contain the following information, the MCQA feature is enabled.



JDBC

You can use the MaxCompute JDBC driver in the following scenarios:

- Use the MaxCompute JDBC driver to connect to MaxCompute. To enable the MCQA feature in this scenario, you must modify related configurations. For more information, see [Enable MCQA for the MaxCompute JDBC driver](#).
- Use the MaxCompute JDBC driver to connect to Tableau. Then, you can use Tableau to analyze data in MaxCompute in a visualized manner. To enable the MCQA feature in this scenario, you must modify related configurations. For more information, see [Enable MCQA on Tableau Server based on the MaxCompute JDBC driver](#).
- Use the MaxCompute JDBC driver to connect to SQL Workbench/J. Then, you can use SQL Workbench/J to execute SQL statements on data in MaxCompute. To enable the MCQA feature in this scenario, you must modify related configurations. For more information, see [Enable MCQA on SQL Workbench/J based on the MaxCompute JDBC driver](#).

Enable MCQA for the MaxCompute JDBC driver

If you use the JDBC driver to connect to MaxCompute, perform the following steps to enable the MCQA feature.

1. Download the [JDBC JAR file](#) that supports the MCQA feature or download the [source code](#) that can be compiled.
2. Add the following dependency to the pom.xml file in the Maven repository:

```
<dependency>
<groupId>com.aliyun.odps</groupId>
<artifactId>odps-jdbc</artifactId>
<version>3.2.0</version>
<classifier>jar-with-dependencies</classifier>
</dependency>
```

Note The version must be V3.2.0 or later.

3. Create a Java program based on the source code and configure the required information. For more information, see [ODPS JDBC](#). The source code contains the following information:

```
String accessId = "your_access_id";
String accessKey = "your_access_key";
String conn = "jdbc:odps:http://service.odps.aliyun.com/api?project=<your_project_name>"&accessId&accessKey&charset=UTF-8&interactiveMode=true";
Statement stmt = conn.createStatement();
-- Replace your_access_id with the AccessKey ID and your_access_key with the AccessKey secret of your Alibaba Cloud account. Replace your_project_name with the name of the project for which the MCQA feature is enabled.
Connection conn = DriverManager.getConnection(conn, accessId, accessKey);
Statement stmt = conn.createStatement();
String tableName = "testOdpsDriverTable";
stmt.execute("drop table if exists " + tableName);
stmt.execute("create table " + tableName + " (key int, value string)");
```

To enable the MCQA feature for the MaxCompute JDBC driver, you need only to modify `String conn` or the code.

- `String conn` : Add `interactiveMode=true` .
 - Code: Add `interactiveMode=true` .
4. (Optional) Configure the following parameters in the connection string to optimize the processing logic.
- `enableOdpsLogger` : is used to display logs. If you do not configure Simple Logging Facade for Java (SLF4J), we recommend that you set this parameter to True.
 - `fallbackForUnknownError` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode when an unknown error occurs.
 - `fallbackForResourceNotEnough` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode when resources are insufficient.
 - `fallbackForUpgrading` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode during an upgrade.
 - `fallbackForRunningTimeout` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode when an operation times out.
 - `fallbackForUnsupportedFeature` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode when the MCQA feature is not supported.
 - `fallbackForAll` : The default value of this parameter is False. If this parameter is set to True, the system rolls back to the offline mode in the preceding scenarios.

Enable MCQA on Tableau Server based on the MaxCompute JDBC driver

Add `interactiveMode=true` to the URL of Tableau Server. We recommend that you also add `enableOdpsLogger=true` to display logs.

Sample URL of Tableau Server:

```
http://service.cn-beijing.maxcompute.aliyun.com/api?project=****_beijing&interactiveMode=true&enableOdpsLogger=true
```

To enable MCQA for some tables in a MaxCompute project, add the `table_list=table_name1,table_name2` property to the URL of Tableau Server. Then, use this property to specify the tables for which you want to enable MCQA. Separate table names with commas (.). If you specify an excessive number of tables, access to the URL of Tableau Server is time-consuming. We recommend that you specify only the required tables in the URL of Tableau Server. Sample URL:

```
http://service.cn-beijing.maxcompute.aliyun.com/api?project=****_beijing&interactiveMode=true&enableOdpsLogger=true&table_list=orders,customers
```

If a table contains a large number of partitions, we recommend that you do not use data from all partitions as the data source. You can filter the required partitions or run custom SQL queries to obtain the required data.

Enable MCQA on SQL Workbench/J based on the MaxCompute JDBC driver

After you configure the MaxCompute JDBC driver, you can use the MCQA feature on SQL Workbench/J by modifying the JDBC URL that you specified on the profile configuration page.

Specify the JDBC URL in the following format: `jdbc:odps:<MaxCompute endpoint>?project=<MaxCompute project name>&accessId=<AccessKey ID>&accessKey=<AccessKey Secret>&charset=UTF-8&interactiveMode=true`. Parameters:

- `maxcompute_endpoint`: the endpoint of your MaxCompute.
- `maxcompute_project_name`: the name of your MaxCompute project.
- `AccessKey ID`: the AccessKey ID that is used to access the specified project.
- `AccessKey Secret`: the AccessKey secret that corresponds to the AccessKey ID.
- `charset=UTF-8`: the character set encoding format.
- `interactiveMode`: specifies whether to enable the MCQA feature. To enable the MCQA feature, set this parameter to `true`.

Enable MCQA by using MaxCompute SDK for Java

You must add a specified dependency to the pom.xml file in the Maven repository. Sample dependency:

```
<dependency>
  <groupId>com.aliyun.odps</groupId>
  <artifactId>odps-sdk-core</artifactId>
  <version>0.36.2-public</version>
</dependency>
```

Run commands to create a Java program. Sample commands:

```
import com.aliyun.odps.Odps;
import com.aliyun.odps.OdpsException;
import com.aliyun.odps.OdpsType;
import com.aliyun.odps.account.Account;
import com.aliyun.odps.account.AliyunAccount;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.data.ResultSet;
import com.aliyun.odps.sqa.*;
import java.io.IOException;
import java.util.*;
public class SQLExecutorExample {
  public static void SimpleExample() {
    // Specify the account and project information.
    Account account = new AliyunAccount("your_access_id", "your_access_key");
    Odps odps = new Odps(account);
    odps.setDefaultProject("your_project_name");
    odps.setEndpoint("http://service.odps.aliyun.com/api");
    // Prepare to build an SQLExecutor.
    SQLExecutorBuilder builder = SQLExecutorBuilder.builder();
    SQLExecutor sqlExecutor = null;
    try {
      // run in offline mode or run in interactive mode
      if (false) {
        // Create an SQLExecutor that runs offline SQL queries by default.
        sqlExecutor = builder.odps(odps).executeMode(ExecuteMode.OFFLINE).build();
      }
    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

```
} else {
    // Create an SQLExecutor that runs SQL queries with MCQA enabled by default.
    sqlExecutor = builder.odps(odps).executeMode(ExecuteMode.INTERACTIVE).build();
}
// Pass special query settings if required.
Map<String, String> queryHint = new HashMap<>();
queryHint.put("odps.sql.mapper.split.size", "128");
// Submit a query job. You can pass hints.
sqlExecutor.run("select count(1) from test_table;", queryHint);
// List the System.out.println() statements that can be used to query common information.
// UUID
System.out.println("ExecutorId:" + sqlExecutor.getId());
// Query the Logview URL of the current query job.
System.out.println("Logview:" + sqlExecutor.getLogView());
// Query the instance on which the current query job is run. In interactive mode, multiple query jobs may be run on the same instance.
System.out.println("InstanceId:" + sqlExecutor.getInstance().getId());
// Query the progress of the current query job. You can check the progress bar in the console.
System.out.println("QueryStageProgress:" + sqlExecutor.getProgress());
// Query the changelogs about the execution status for the current query job, such as rollback messages.
System.out.println("QueryExecutionLog:" + sqlExecutor.getExecutionLog());
// Obtain results of query jobs by calling one of the following API operations:
if(false) {
    // Query the results of all query jobs. The API operation that you called is a synchronous operation and may occupy a thread until the query succeeds or fails.
    List<Record> records = sqlExecutor.getResult();
    printRecords(records);
} else {
    // Query the ResultSet iterator of the query results. The API operation that you called is a synchronous operation and may occupy a thread until the query succeeds or fails.
    ResultSet resultSet = sqlExecutor.getResultSet();
    while (resultSet.hasNext()) {
        printRecord(resultSet.next());
    }
}
// run another query
sqlExecutor.run("select * from test_table;", new HashMap<>());
if(false) {
    // Query the results of all query jobs. The API operation that you called is a synchronous operation and may occupy a thread until the query succeeds or fails.
    List<Record> records = sqlExecutor.getResult();
    printRecords(records);
} else {
    // Query the ResultSet iterator of the query results. The API operation that you called is a synchronous operation and may occupy a thread until the query succeeds or fails.
    ResultSet resultSet = sqlExecutor.getResultSet();
    while (resultSet.hasNext()) {
        printRecord(resultSet.next());
    }
}
} catch (OdpsException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
} finally {
    if (sqlExecutor != null) {
        // Close the SQLExecutor and release related resources.
        sqlExecutor.close();
    }
}
```

```

        sqlExecutor.close();
    }
}
// SQLExecutor can be reused by pool mode
public static void ExampleWithPool() {
    // Specify the account and project information.
    Account account = new AliyunAccount("your_access_id", "your_access_key");
    Odps odps = new Odps(account);
    odps.setDefaultProject("your_project_name");
    odps.setEndpoint("http://service.odps.aliyun.com/api");
    // Run queries by using a connection pool.
    SQLExecutorPool sqlExecutorPool = null;
    SQLExecutor sqlExecutor = null;
    try {
        // Create a connection pool. Specify the connection pool size and the default execution mode.
        SQLExecutorPoolBuilder builder = SQLExecutorPoolBuilder.builder();
        builder.odps(odps)
            .initPoolSize(1) // init pool executor number
            .maxPoolSize(5) // max executors in pool
            .executeMode(ExecuteMode.INTERACTIVE); // run in interactive mode
        sqlExecutorPool = builder.build();
        // Obtain an SQLExecutor from the connection pool. If no SQLExecutor can be obtained from the connection pool, you can add an SQLExecutor. Make sure that the total number of SQLExecutors does not exceed the upper limit.
        sqlExecutor = sqlExecutorPool.getExecutor();
        // Use the SQLExecutor in the same way it is used in the preceding example.
        sqlExecutor.run("select count(1) from test_table;", new HashMap<>());
        System.out.println("InstanceId:" + sqlExecutor.getId());
        System.out.println("Logview:" + sqlExecutor.getLogView());
        List<Record> records = sqlExecutor.getResult();
        printRecords(records);
    } catch (OdpsException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        sqlExecutor.close();
    }
    sqlExecutorPool.close();
}
private static void printRecord(Record record) {
    for (int k = 0; k < record.getColumnCount(); k++) {
        if (k != 0) {
            System.out.print("\t");
        }
        if (record.getColumns()[k].getType().equals(OdpsType.STRING)) {
            System.out.print(record.getString(k));
        } else if (record.getColumns()[k].getType().equals(OdpsType.BIGINT)) {
            System.out.print(record.getBigint(k));
        } else {
            System.out.print(record.get(k));
        }
    }
}
private static void printRecords(List<Record> records) {
    for (Record record : records) {
        printRecord(record);
    }
}

```

```
        System.out.println();
    }
}
public static void main(String args[]) {
    SimpleExample();
    ExampleWithPool();
}
}
```

1.23.3. VVP On MaxCompute

1.23.3.1. Overview

This topic describes VVP On MaxCompute in MaxCompute feature enhancement packages.

VVP On MaxCompute encapsulates the features of Realtime Compute for Apache Flink that is developed on the Ververica Platform (VVP) based on MaxCompute resources. After you enable the VVP On MaxCompute feature, you can use the Cupid joint computing platform to complete the operations related to real-time computing by using the underlying storage and computing resources of MaxCompute on the VVP UI.

1.23.3.2. Benefits

This topic describes the benefits of VVP On MaxCompute.

- Seamless integration with MaxCompute
 - MaxCompute is interconnected with Realtime Compute for Apache Flink to share data. This way, Realtime Compute for Apache Flink can benefit from the scheduling and storage capabilities of MaxCompute.
 - Computing and storage resources are managed in a unified manner to ensure high elasticity. You do not need to be concerned about the issues, such as storage space expansion difficulties and time-consuming data computations that are caused by the increased data volume. The storage and retrieval capabilities of a MaxCompute cluster are automatically extended with your data volume. This enables you to focus on your business and data value.
 - Hardware integration is used to eliminate the differences in machine configuration requirements, time consumption, and peak loads of stream and batch processing. This way, the number of machines in all clusters is reduced for lower hardware costs and the model is unified for lower maintenance costs.
 - Intelligent cluster deployment and O&M of Realtime Compute for Apache Flink: You can deploy Realtime Compute for Apache Flink along with MaxCompute to achieve real-time data processing and harness all the capabilities provided by MaxCompute and its related ecosystem components, such as Spark, Elasticsearch, advanced data analysis, and AI.
 - Enterprise-class system security: Security isolation and control are implemented based on the multi-tenancy capability and security control of MaxCompute.
 - APIs: SDK programming interfaces for Apache Flink are used to support features provided by Apache Flink and user-defined features.
- End-to-end platform development: Jobs can be created, configured, and run on MaxCompute.
- End-to-end platform O&M: Comprehensive monitoring of job running status and optimization and troubleshooting of MaxCompute are implemented. Enterprise-class services are provided, and service issues can be troubleshot and optimized.
- Excellent platform ecosystem: Real-time data computing and batch processing of data warehouses are supported on the same resource platform to integrate batch and stream processing.
- Full compatibility with enterprise-class computing engines of Apache Flink

- Business GeminiStateBackend is provided, which brings the following benefits:
 - Uses a new data structure to accelerate ad hoc queries and reduce frequent disk I/O operations.
 - Optimizes the cache policy. If memory is sufficient, hot data is not stored in disks and cache entries do not expire after compaction.
 - Uses Java to implement GeminiStateBackend, which eliminates Java Native Interface (JNI) overheads that are caused by RocksDB.
 - Uses off-heap memory and implements an efficient memory allocator based on GeminiDB to eliminate the impact of garbage collection for Java Virtual Machines (JVMs).
 - Supports asynchronous incremental checkpoints. This ensures that only memory indexes are copied during data synchronization. Compared with RocksDB, GeminiStateBackend avoids jitters that are caused by I/O operations.
 - Supports local recovery and storage of the timer.
- Support for various connectors: Connectors for services, such as Log Service, DataHub, Message Queue for Apache Kafka, and Tablestore, are supported. The supported connectors are continuously updated.
- Full compatibility with Apache Flink: VVP On MaxCompute is continuously updated along with release updates of Apache Flink.

1.23.3.3. Activate VVP On MaxCompute

This topic describes how to activate VVP On MaxCompute.

 **Note** You must perform the following operations as user admin, unless otherwise specified.

1. Run the following command on ODPS AG to start the Webrm service:

```
cd /home/admin/vvp/web_rm_server
sudo sh webrm.sh restart
```

After you run the command, run `netstat -nlp | grep 9088`. If the following information is displayed, the Webrm service is started.

```
$netstat -nlp | grep 9088
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 10.5.10.203:9088      0.0.0.0:*              LISTEN      -
```

2. Run the following command on ODPS AG to check the running status of Docker containers.

```
ps -ef | grep vvp
```

```
$docker ps | grep vvp
4d6e2222fa99      39c894ad404b      "tail -f /dev/null"      2 days ago      Up 2 days
    odps-service-regress.VVPUI__vvp_ui.1606281917
73e611f482b8      8d34715b6d78      "tail -f /dev/null"      2 days ago      Up 2 days
    odps-service-regress.VVPAppmanager__vvp_appmanager.1606281916
9b3b7ed738a6      d2aa039a3150      "tail -f /dev/null"      2 days ago      Up 2 days
    odps-service-regress.VVPGateway__vvp_gateway.1606281914
```

3. Log on to the `vvp_ui` container based on the container ID obtained in Step 2. Go to the `/home/admin` directory, and then run the `./start` command to run the script. If you run `netstat -nlp | grep 4200` in a new window on ODPS AG and the following information is displayed, the container starts.

```
$netstat -nlp | grep 4200
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:4200          0.0.0.0:*              LISTEN      -
```

4. Log on to the `vp_gateway` container based on the container ID obtained in Step 2. Go to the `/home/admin` directory, and then run the `./start` command to run the script. If you run `netstat -nlp|grep 8989` in a new window on ODPS AG and the following information is displayed, the container starts.

```
$netstat -nlp|grep 8989
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:8989          0.0.0.0:*           LISTEN      -
```

5. Log on to the `vp_appmanager` container based on the container ID obtained in Step 2. Go to the `/home/admin` directory, and then run the `./start` command to run the script. If you run `netstat -nlp|grep 9100` in a new window on ODPS AG and the following information is displayed, the container starts.

```
$netstat -nlp|grep 9100
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:9100         0.0.0.0:*           LISTEN      -
```

Note Perform Step 3 to Step 5 in sequence. The next step can be performed only after the preceding step is complete. Random order is not allowed.

VVP On MaxCompute is activated. You can complete relevant operations on the VVP UI.

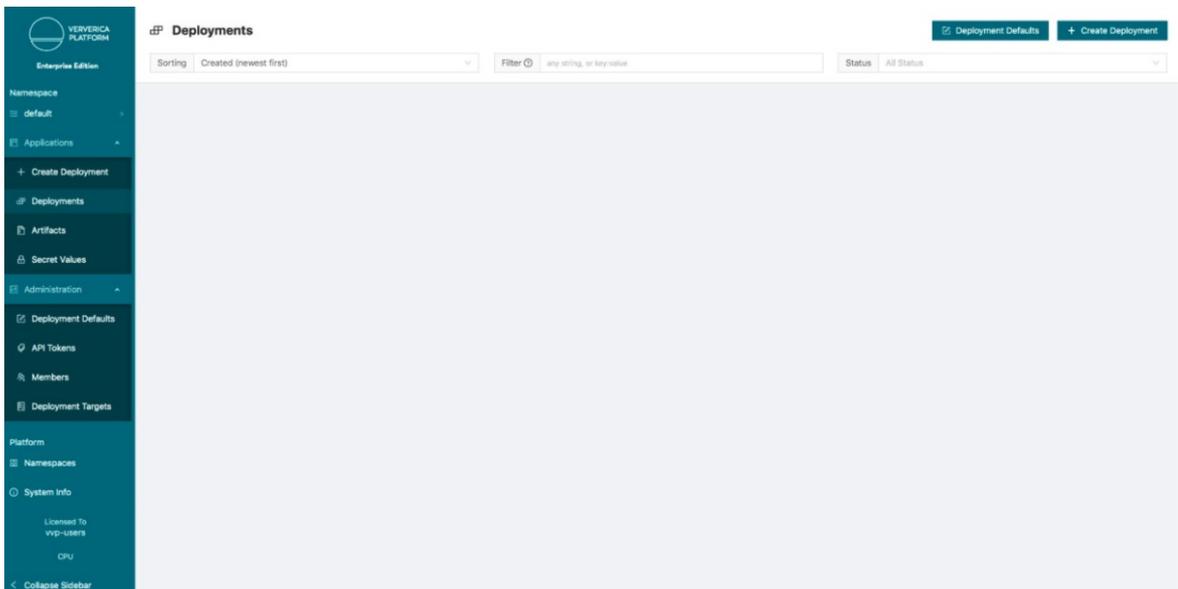
1.23.3.4. Usage notes

This topic describes how to perform operations on the VVP UI.

1. Open your browser, enter the URL of the VVP UI in the address bar, and then press Enter to log on to the VVP UI.

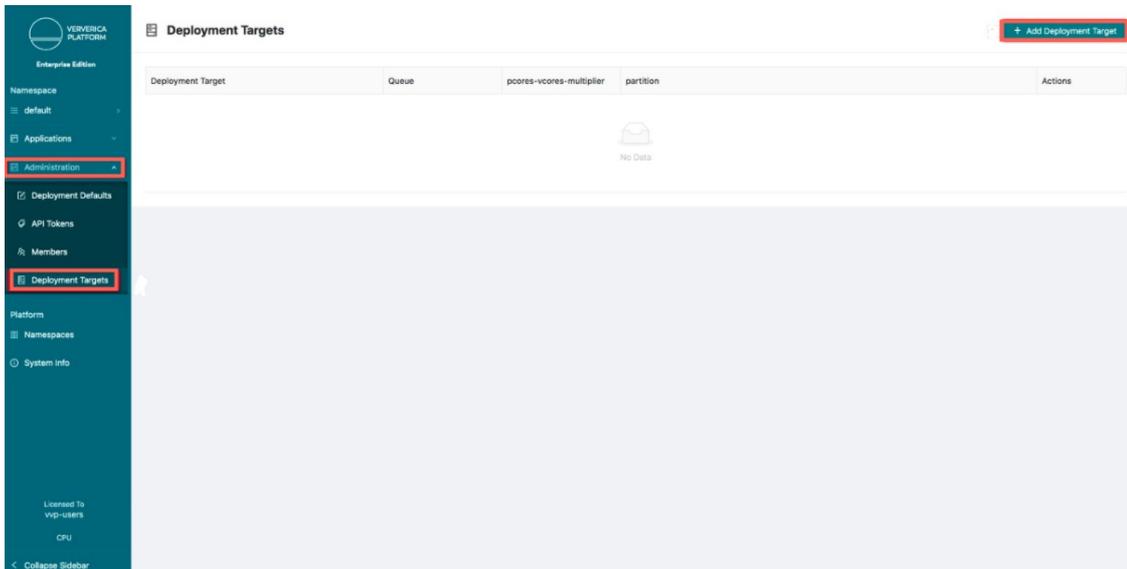
The URL of the VVP UI is in the format of `xx.xx.xx.xx:8989`. `xx.xx.xx.xx` indicates the IP address of ODPS AG and `8989` is the port number.

Note We recommend that you use the Google Chrome browser.



2. Create a deployment target.

- i. In the left-side navigation pane, choose **Administration > Deployment Targets**. On the Deployment Targets page, click **+ Add Deployment Target**. The Add Deployment Target dialog box appears.



- ii. In the Add Deployment Target dialog box, specify Deployment Target Name and Yarn Queue, and click **OK**.

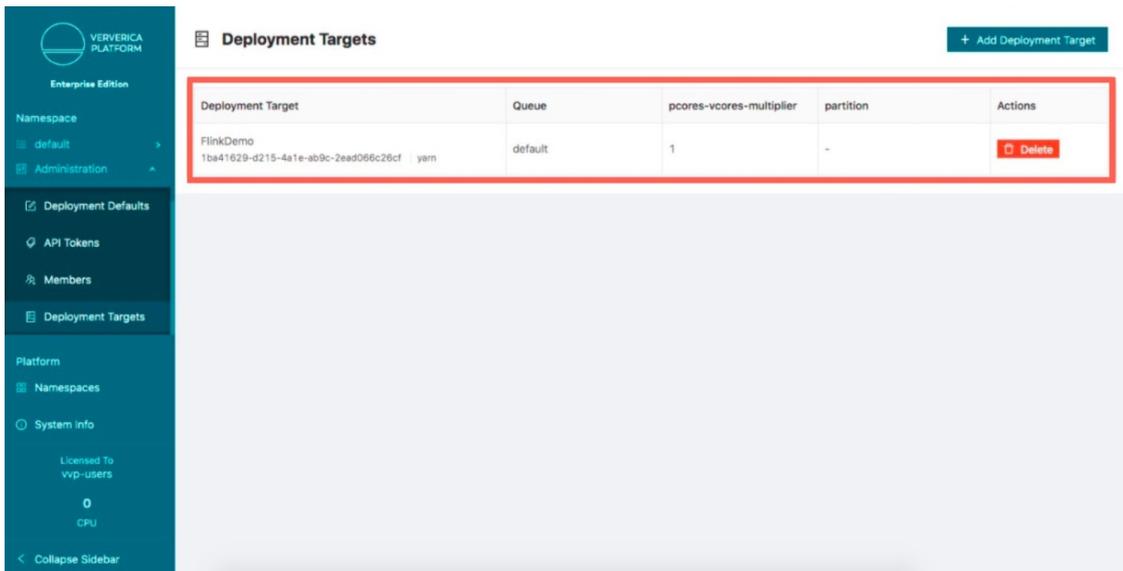
Note Deployment Target Name allows you to customize the settings. Yarn Queue must be set to default.

The 'Add Deployment Target' dialog box is shown with the following fields and values:

- Deployment Target Name:** FlinkDemo
- Yarn Queue:** default
- pcores-voores-multiplier:** 1
- partition:** partition

A yellow warning message states: "It is not recommended to use the same Yarn Queue for VeriData Platform and your Deployments." The dialog includes 'Cancel' and 'OK' buttons at the bottom.

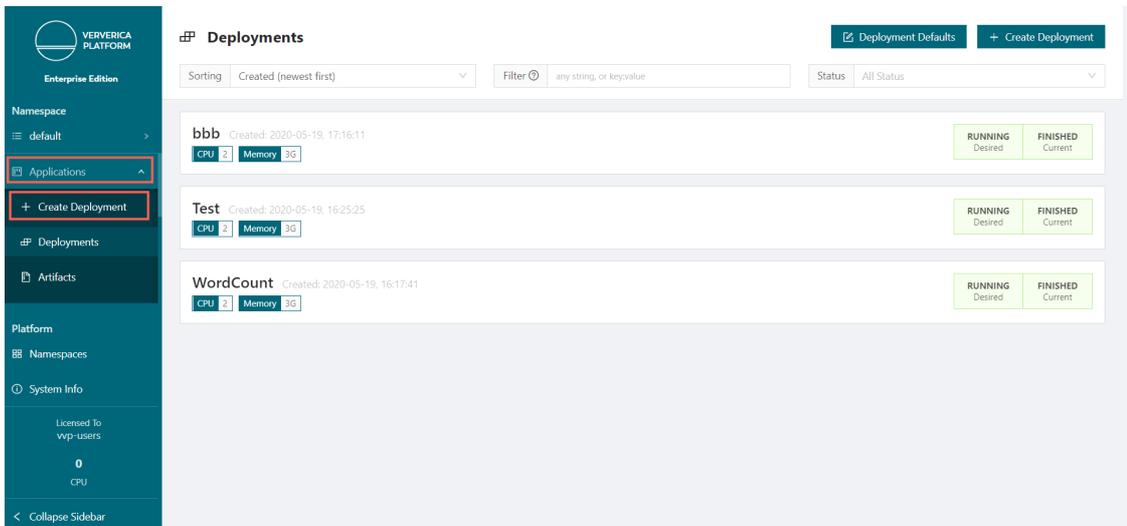
iii. View the information of the deployment target that you created on the VVP UI.



3. Create a deployment.

Note Before you create a deployment, make sure that the deployment target already exists.

i. In the left-side navigation pane, choose **Applications > + Create Deployment**. The **Create Deployment** page appears.



- ii. Click the **Advanced** tab and configure parameters.

Note Deployment Name and Labels allow you to customize settings. The value of Labels is in the format of key-value pairs. Select an existing deployment target from the Deployment Target drop-down list, enter the address for downloading the JAR file in Jar URI, and then retain default values for other parameters.

Create Deployment

Standard **Advanced** YAML

Organization

* Deployment Name Non-Production Mode

Please enter a name for this Deployment

Labels

Label key Label value

Behavior

Deployment Target

Configuration

* Jar URI

- iii. Click **Create Deployment** to create the deployment.

Logging

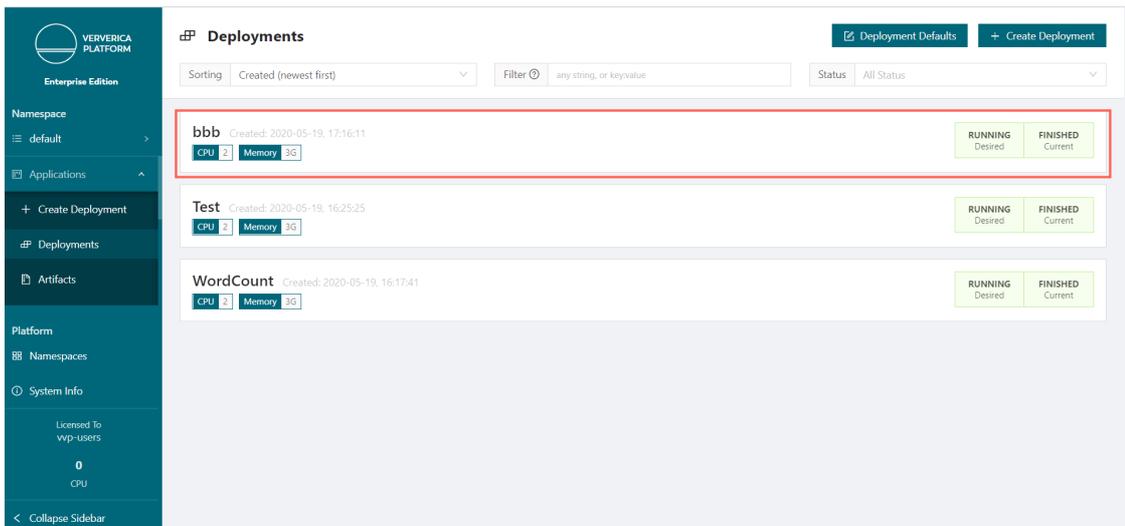
Root Log Level

Log Levels

Logger name Log level

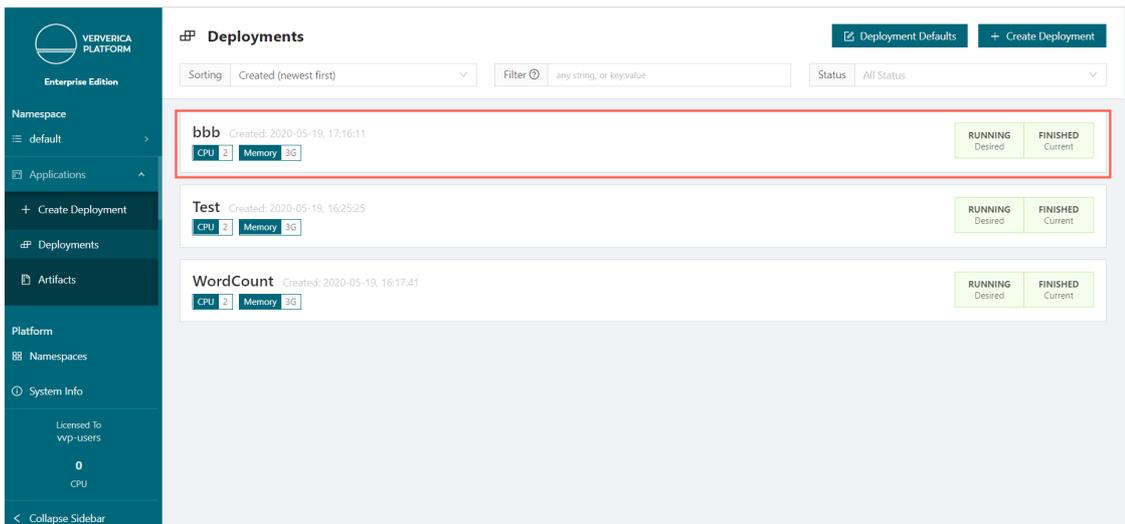
Logging Profile

iv. View the information of the new deployment on the VVP UI.

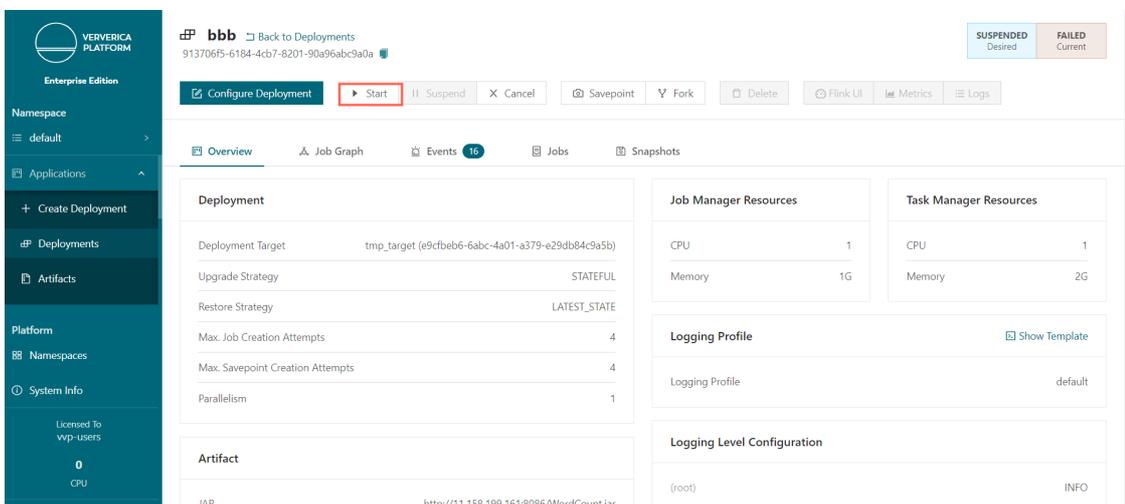


4. Start the deployment.

- i. In the left-side navigation pane, choose **Applications > Deployments**. On the Deployments page, click the deployment that you created. The deployment details page appears.

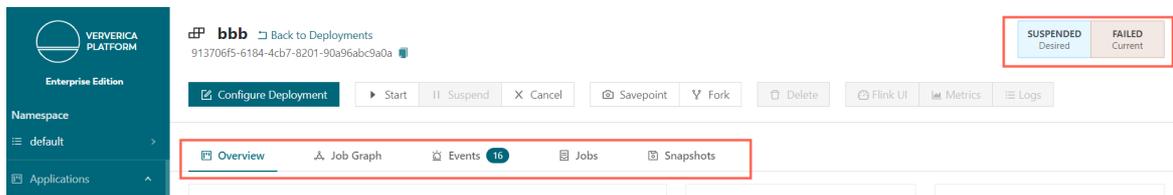


- ii. On the deployment details page, click **Start** to start the deployment.

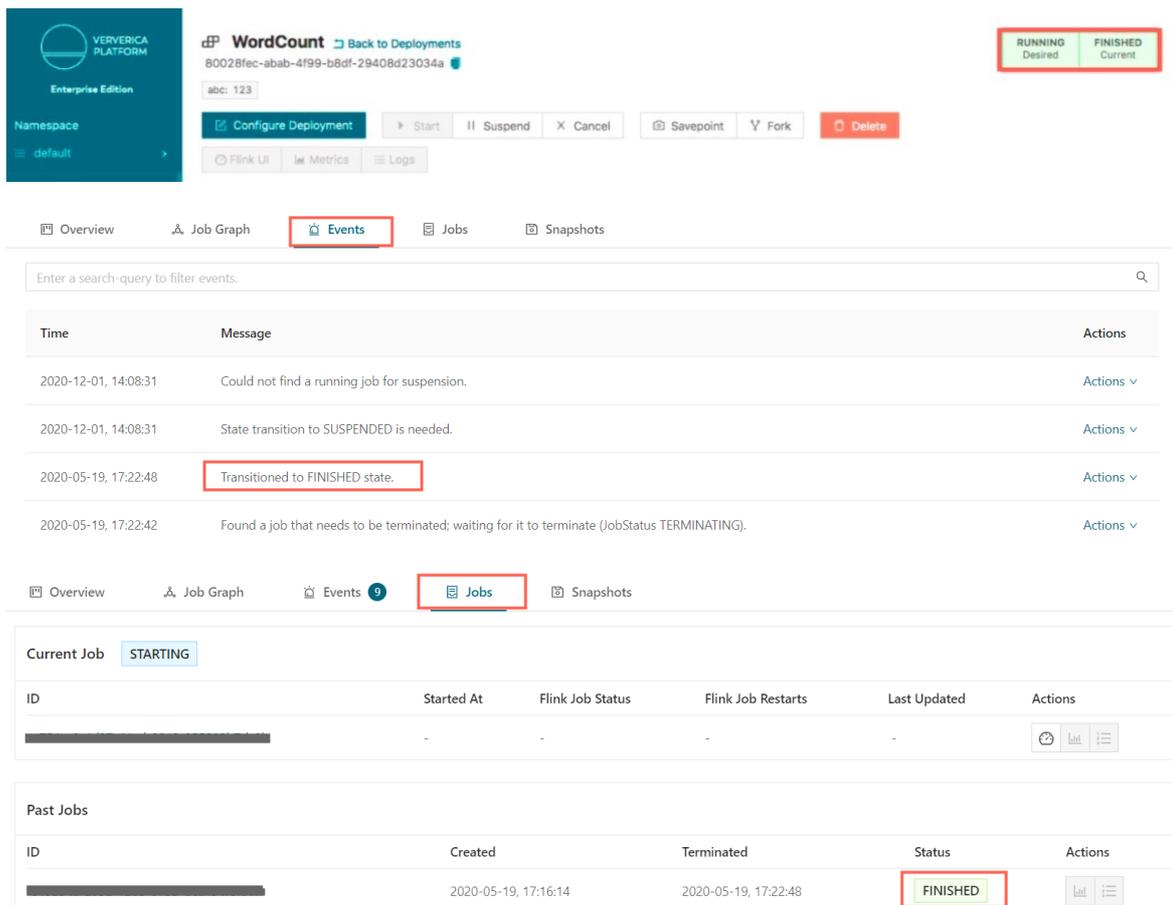


5. View the information of the deployment.

When you run the deployment, you can view the information shown on the deployment details page. The information provides the running status of the deployment.



After the deployment is complete, its status changes. You can view the deployment completion information on the **Events** and **Jobs** tabs.



If an error occurs on a running deployment, you can view further details about error information on the **Events** tab.



1.24. MaxCompute FAQ

This topic provides answers to some frequently asked questions about MaxCompute.

How do I check MaxCompute resource usage when SQL statements are slowly executed?

Log on to the host where MaxCompute AG is deployed as user admin and perform the following steps:

1. Run the following command to sort the hosts in ascending order of the number of remaining resources on each host:

```
rtfrr|sed 's,/,/g'|sort -t "|" -k2 -n
```

2. Run the following command to view resource details of each host and total cluster resources in MaxCompute:

```
rttr|sed 's,/,/g'
```

3. Calculate the percentage of remaining resources to all resources in the cluster based on the statistics obtained in the preceding steps. Then, obtain the resource usage of MaxCompute.

Some jobs submitted by a project are slowly executed even if remaining resources in a MaxCompute cluster are sufficient. What do I do?

A possible cause is that the resources for the quota group where the project resides are exhausted. Perform the following steps to check whether the resources are exhausted and determine whether to add resources to the quota group:

1. Log on to the host where MaxCompute AG is deployed as user admin and run the following command to check the resource usage of the quota group:

```
r quota
```

2. If resources in the quota group are exhausted, you can use Apsara Big Data Manager (ABM) to modify the settings of the quota group.

How do I modify the settings of a quota group?

1. Log on to the host where MaxCompute AG is deployed as user admin and run the following command to create or modify quotas for the quota group.

```
sh/apsara/deploy/rpc_wrapper/rpc.sh setquota -i $QUOTAID -a $QUOTANAME -t fair -s$max_cpu_quota $max_mem_quota -m $min_cpu_quota $min_mem_quota
```

 **Note** If \$QUOTAID exists, the quotas are modified. Otherwise, quotas are created.

2. Log on to ABM to configure related settings.

How do I perform simple operations on the metadata warehouse?

1. Log on to the host where MaxCompute AG is deployed as user admin.
2. Run the following commands:

```
/apsara/odps_tools/clt/bin/odpscmd
```

```
use meta;
```

3. Run the following command to view all tables in the metadata warehouse:

```
show tables;
```

4. Run the following command to obtain the description of a specific table:

```
desc <table>;
```

How do I use the smart metadata warehouse?

You must install the metadata warehouse enhancement package `package+view` first. Before you install the package, make sure that you are the owner of a project that is granted the package installation permission. After you install the package, you can use the smart metadata warehouse. For more information, see [Operations for package creators](#).

• Installation

`package+view` is a metadata warehouse enhancement package. You can run the `odpscmd --config=odps_config.ini -f init.sql` command in the *system metadata warehouse* directory to complete the installation.

ⓘ Note

- If the system asks you to re-install the package, comment out `create package` from the first line of `odpscmd --config=odps_config.ini -f init.sql`, and then run the command again.
- `odps_config.ini` is a configuration file that contains the configuration of the account. The account you configured is also the project owner of the metadata warehouse and can access the metadata warehouse.

• Permission assignment

Run the following command to allow you to install the `package+view` package for a project, such as the `p1` project:

```
odpscmd --config=odps_config.ini -e "allow project p1 to install package systables;"
```

Run the following command to allow you to install the `package+view` package for all projects:

```
odpscmd --config=odps_config.ini -e "allow project * to install package systables;"
```

• User operations

You can run the following command in `odpscmd` to install the package. Before you perform this step, make sure that you are the owner of the project that is granted the package installation permission.

```
install package meta.systables;
```

After the installation is complete, you can run the following command to query the description of views in the package:

```
desc package meta.systables;
```

ⓘ Note After the preceding operations are complete, you can use the smart metadata warehouse.

• Views

To learn more about the definition of the table schema, see the related content in the view description, which is displayed after you run the following command:

```
desc viewname
```

View name	Content
<code>allowed_package_installers</code>	Information about the project that is granted the package installation permission
<code>column_label_grants</code>	Column label authorization information

View name	Content
column_labels	Column label information of a table
columns	Table schema information
installed_packages	Information about the package installed for the project
object_privileges	Authorization information of tables, UDFs, and resources
package_resources	Object information contained in the package
partitions	Partition information of a partitioned table
policies	Policies that define user or role permissions
resources	Resource information
roles	Role information
table_label_grants	Label authorization information of a table
table_labels	Label information of a table
tables	Table or view information
tasks	Job execution records
tunnels	Data upload and download records
udf_resources	Information of resources used in UDFs
udfs	Information of UDFs
user_roles	Information of associations between users and roles
users	User information

● **Notes:**

- By default, the package+view package only allows you to query metadata in the last 180 days.
- If you do not run a job on a day, the metadata warehouse does not store the metadata of the day.
- We recommend that you specify a query range to prevent a full data scan in the last 180 days.
- The metadata of the previous day can be queried only after the metadata is generated by the metadata warehouse in the early morning of the next day. In the current configuration, metadata can be queried immediately after it is generated by the metadata warehouse.
- The metadata warehouse does not provide metadata on the day when a project is created. The purpose is to obtain only the metadata of the projects with unique names.

How do I grant Java sandbox permissions?

1. Log on to AdminConsole and choose **MaxCompute Configuration > Project Management**. Select the project to which you want to grant Java sandbox permissions and double-click the project to open the property dialog box.

2. In the **ODPS Sandbox Setting** section, enter the method or class that you want to use in Sandbox Java Permissions.

 **Note** Make sure that the content you entered is in a valid format. The following example is for reference only. Enter each item in a single line and end it with a semicolon (;).

```
permission java.lang.RuntimePermission "readSystemProperty"; permission java.lang.RuntimePermission "modifyThreadGroup"; permission java.security.AllPermission;
```

3. Click **Finish Modification**.

What do I do if disk space is insufficient?

In most cases, you can delete scripts to release the disk space. The most possible cause is that the root directory of MaxCompute AG or the /apsara directory occupies too much disk space. Therefore, you must delete scripts in the two directories.

How do I add a MaxCompute host to a blacklist?

1. Log on to Apsara AG as user admin. Run the following command to enable the Fuxi blacklisting function:

```
rsgf fuximaster>{"fuxi_Enable_BadNodeManager":false}"
```

2. Run the following command to view the Fuxi blacklist:

```
/apsara/deploy/rpc_wrapper/rpc.shblacklist cluster get
```

3. Run the following command to add the host to the Fuxi blacklist:

```
/apsara/deploy/rpc_wrapper/rpc.shblacklist cluster add $hostname
```

4. Run the following command to check that the host has been added to the Fuxi blacklist:

```
/apsara/deploy/rpc_wrapper/rpc.shblacklist cluster get
```

How do I export data from MaxCompute?

You can use a Tunnel command to export data. You can also configure synchronization tasks in DataWorks to export data from MaxCompute to other destinations.

How do I view the MaxCompute version?

Run the following commands to obtain the version of MaxCompute:

```
cat /apsara/odps_info/version|grep odps
```

```
cat /apsara/version
```

How do I restart services in MaxCompute?

1. Run the following commands to save the configurations of resident services of MaxCompute to a file. This configuration file is required when you restart MaxCompute services.

```
ssh odpsAG
cd /home/admin/
# If you do not use a service, you can ignore the command that is not required.
# You can run the r al command to view resident services.
r plan Odps/CGServiceControllerx > CGServiceControllerx
r plan sys/sqlonline-OTS > sqlonline-OTS
r plan Odps/MessengerService > MessengerService
r plan Odps/OdpsService > OdpsService
r plan Odps/HiveServerx > HiveServerx
r plan Odps/XStreamService > XStreamService
r plan Odps/QuotaService > QuotaService
r plan Odps/ReplicationService > ReplicationService
```

2. Run the following commands to stop services in MaxCompute:

```
r sstop Odps/CGServiceControllerx
r sstop sys/sqlonline-OTS
r sstop Odps/MessengerService
r sstop Odps/OdpsService
r sstop Odps/HiveServerx
r sstop Odps/XStreamService
r sstop Odps/QuotaService
r sstop Odps/ReplicationService
```

3. Run the following commands to start services in MaxCompute:

```
ssh odpsAG
cd /home/admin/
r start CGServiceControllerx
r start sqlonline-OTS
r start MessengerService.txt
r start OdpsService.txt
r start HiveServerx.txt
r start XStreamService.txt
r start QuotaService.txt
r start ReplicationService.txt
```

How do I power off MaxCompute and then power it on?

1. Run the following commands to save the configurations of resident services of MaxCompute to a file. This configuration file is required when you restart MaxCompute services.

```
ssh odpsAG
cd /home/admin/
# If you do not use a service, you can ignore the command that is not required.
# You can run the r al command to view resident services.
r plan Odps/CGServiceControllerx > CGServiceControllerx
r plan sys/sqlonline-OTS > sqlonline-OTS
r plan Odps/MessengerService > MessengerService
r plan Odps/OdpsService > OdpsService
r plan Odps/HiveServerx > HiveServerx
r plan Odps/XStreamService > XStreamService
r plan Odps/QuotaService > QuotaService
r plan Odps/ReplicationService > ReplicationService
```

2. Run the following commands to stop services in MaxCompute:

```
r sstop Odps/CGServiceControllerx  
r sstop sys/sqlonline-OTS  
r sstop Odps/MessengerServicex  
r sstop Odps/OdpsServicex  
r sstop Odps/HiveServerx  
r sstop Odps/XStreamServicex  
r sstop Odps/QuotaServicex  
r sstop Odps/ReplicationServicex
```

3. Run the following command to shut down the Apsara distributed operating system:

```
/home/admin/dayu/bin/allapsara stop
```

4. Run the following command to shut down compute nodes gracefully:

```
Shutdown
```

5. Start compute nodes.
6. Run the following command to start the Apsara distributed operating system:

```
/home/admin/dayu/bin/allapsara start
```

7. Run the following commands to start services in MaxCompute:

```
ssh odpsAG  
cd /home/admin/  
r start CGServiceControllerx  
r start sqlonline-OTS  
r start MessengerServicex.txt  
r start OdpsServicex.txt  
r start HiveServerx.txt  
r start XStreamServicex.txt  
r start QuotaServicex.txt  
r start ReplicationServicex.txt
```

How do I reduce the heavy load on a host?

1. Log on to the host with a heavy load and run the `top` command to check whether task processes occupy a large number of resources.
2. If the check result shows that task processes occupy a large number of resources, terminate the tasks after these tasks are complete or after communication with users.

1.25. Open source features of MaxCompute

This topic describes open source features of MaxCompute.

SDK

MaxCompute provides APIs for SDK for Java and SDK for Python to create, view, and delete MaxCompute tables. You can use SDKs to edit code and perform required operations on MaxCompute.

Technical support: View the official documentation or submit a ticket.

MaxCompute RODPS

MaxCompute RODPS is an R plug-in for MaxCompute. For more information, see [aliyun-odps-r-plugin](#) on GitHub.

Technical support: Leave a message or create an issue in [aliyun-odps-r-plugin](#) on GitHub.

MaxCompute JDBC

MaxCompute JDBC is an official JDBC driver provided by MaxCompute. It provides a set of interfaces to run SQL tasks for Java programs. The project is hosted in [aliyun-odps-jdbc](#) on GitHub.

Technical support: Leave a message or create an issue in [aliyun-odps-jdbc](#) on GitHub.

Data Collector

Data Collector is a collection of the major open source data collection tools of MaxCompute, such as the Flume plug-in, OGG plug-in, Sqoop, Kettle plug-in, and Hive Data Transfer UDTF.

Flume and OGG plug-ins are implemented based on the DataHub SDK, whereas Sqoop, Kettle plug-in, and Hive Data Transfer UDTF are implemented based on the Tunnel SDK. DataHub is a real-time data transfer channel, and Tunnel is a batch data transfer channel. The Flume and OGG plug-ins are used to transfer data in real time. Sqoop, Kettle plug-in, and Hive Data Transfer UDTF are used to transfer data in offline mode.

For information about the source code, see [aliyun-maxcompute-data-collectors](#) on GitHub. For more information about these tools, see [wiki](#) on GitHub.

Technical support: Leave a message or create an issue in [aliyun-maxcompute-data-collectors](#) on GitHub.

2.DataWorks

2.1. Log on to the DataWorks console

This topic describes how to log on to the DataWorks console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

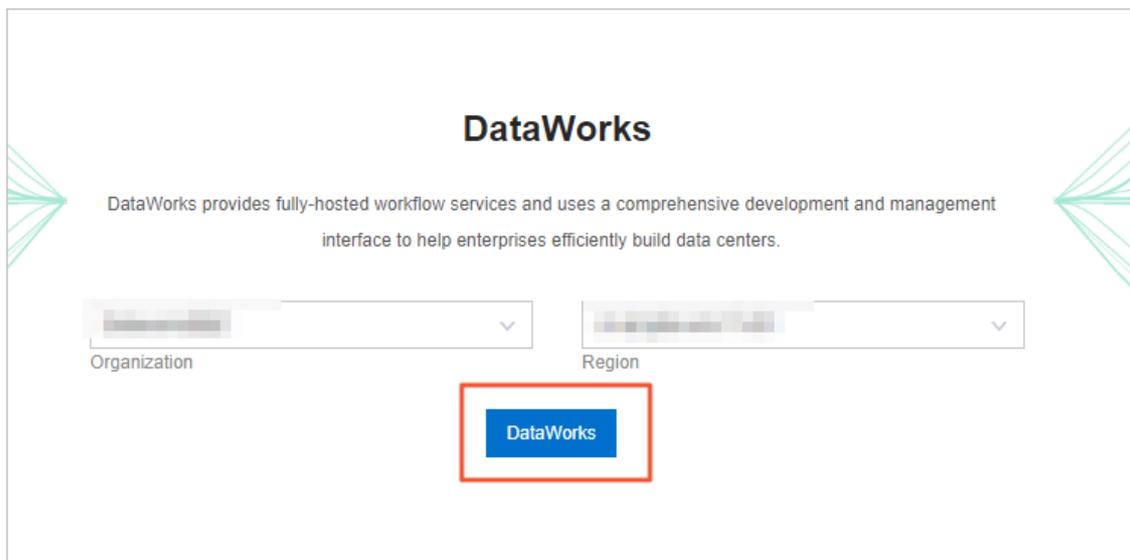
1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

Note When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, choose **Products > Big Data > DataWorks**.
5. On the page that appears, set the **Organization** and **Region** parameters and click **DataWorks**.



 **Notice** You are not allowed to log on to the DataWorks console by using the root organization.

2.2. Create a workspace

This topic describes how to create a workspace on the Project Management page.

Prerequisites

A compute engine is created to initialize MaxCompute projects.

Overview

DataWorks provides various preset templates for a workspace administrator to select when the administrator creates workspaces that contain one or more working environments, including development, testing, staging, and production. DataWorks can also automatically generate associations between workspaces. A one-to-many relationship exists between departments and workspaces. That is, multiple workspaces can be created under a department.

You can create a workspace in one of the following modes:

- **Standard Mode (Development and Production Environments):** A DataWorks workspace corresponds to two MaxCompute projects. One MaxCompute project serves as the development environment and the other serves as the production environment.
- **Basic Mode (Production Environment Only):** A DataWorks workspace corresponds to only one MaxCompute project.

 **Note** For more information about the two workspace modes, see [Workspace modes](#).

Procedure

1. Log on to the DataWorks console as a workspace administrator.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Project Management**. By default, the **Workspaces** page appears.
3. On the Workspaces page, click **Create Workspace** in the upper-right corner.
4. In the **Create Workspace** dialog box that appears, set the parameters in the Basic Information section.

 **Note** If you select the standard mode, you must associate the workspace with two MaxCompute projects.

5. Set the parameters in the **Advanced Settings** section. You can select whether to enable the recurrence and whether to allow downloading the query results returned by SELECT statements. You must associate the workspace with MaxCompute projects.
6. Click **OK**.

2.3. Quick Start

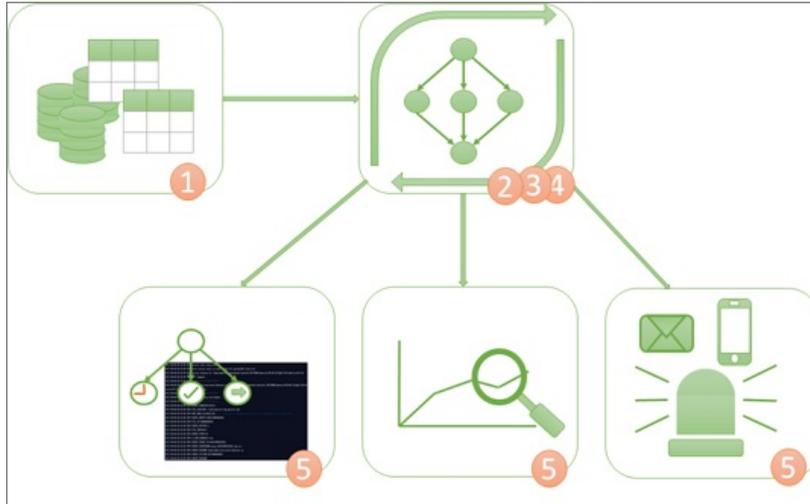
2.3.1. Overview

Quick Start guides you through a complete process of data analytics and O&M.

Generally, you can complete the following data analytics and O&M operations in a workspace of DataWorks:

1. Create tables and import data.
2. Create a workflow.
3. Create a sync node.
4. Configure recurrence and dependencies for a node.
5. Run a node and troubleshoot errors.

The following figure shows the basic process of data analytics and O&M.



2.3.2. Create tables and import data

This topic takes the `bank_data` and `result_table` tables as an example to describe how to create tables and import data in the DataWorks console.

Note The `bank_data` table stores business data, whereas the `result_table` table stores data analytics results.

Create the `bank_data` table

1. Log on to the DataWorks console.
2. On the **DataStudio** page that appears, move the pointer over the  icon and click **Table**.
3. In the **Create Table** dialog box, set **Table Name** to `bank_data`.
4. Click **Commit**.
5. On the editing page of the created table, click **DDL Statement**.
6. In the **DDL Statement** dialog box, enter the table creation statement, and click **Generate Table Schema**. In the dialog box that appears, click **OK**.

In this topic, the following statement is used as an example:

```
CREATE TABLE IF NOT EXISTS bank_data
(
  age      BIGINT COMMENT 'age',
  job      STRING COMMENT 'job type',
  marital  STRING COMMENT 'marital status',
  education STRING COMMENT 'education level',
  default  STRING COMMENT 'credit card',
  housing  STRING COMMENT 'mortgage',
  loan     STRING COMMENT 'loan',
  contact  STRING COMMENT 'contact',
  month    STRING COMMENT 'month',
  day_of_week STRING COMMENT 'day in a week',
  duration STRING COMMENT 'duration',
  campaign BIGINT COMMENT 'number of contacts during the campaign',
  pdays    DOUBLE COMMENT 'interval from the last contact',
  previous DOUBLE COMMENT 'number of contacts with the customer',
  poutcome STRING COMMENT 'result of the previous marketing campaign',
  emp_var_rate DOUBLE COMMENT 'employment change rate',
  cons_price_idx DOUBLE COMMENT 'consumer price index',
  cons_conf_idx DOUBLE COMMENT 'consumer confidence index',
  euribor3m DOUBLE COMMENT 'Euro deposit rate',
  nr_employed DOUBLE COMMENT 'number of employees',
  y        BIGINT COMMENT 'whether time deposit is available'
);
```

7. After the table schema is generated, enter the display name of the table and click **Commit to Development Environment** or **Commit to Production Environment**.

 **Note** If you are using a workspace of the basic mode, click **Commit to Production Environment**.

8. In the left-side navigation pane, click **Workspace Tables**. On the page that appears, enter the table name to search for the created table. After you find the table, double-click the table name to view the table information.

Create the result_table table

1. On the **DataStudio** page that appears, move the pointer over the  icon and click **Table**.
2. In the **Create Table** dialog box, set **Table Name** to result_table.
3. On the editing page of the created table, click **DDL Statement**.
4. In the **DDL Statement** dialog box, enter the table creation statement, and click **Generate Table Schema**. In the dialog box that appears, click **OK**.

In this topic, the following statement is used as an example:

```
CREATE TABLE IF NOT EXISTS result_table
(
  education STRING COMMENT 'education level',
  num       BIGINT COMMENT 'number of people'
);
```

5. After the table schema is generated, enter the display name of the table and click **Commit to Development Environment** or **Commit to Production Environment**.
6. In the left-side navigation pane, click **Workspace Tables**. On the page that appears, enter the table name to search for the created table. After you find the table, double-click the table name to view the

table information.

Upload a local file to import its data to the bank_data table

You can perform the following operations in the DataWorks console:

- Upload a local text file to import its data to a table in a workspace.
- Use Data Integration to import business data from different data stores to a workspace.

Note In this topic, a local file is used as the source of data. Comply with the following rules when uploading a local file:

- File format: The file must be in the .txt, .csv, or .log format.
- File size: The size of the file cannot exceed 10 MB.
- Destination object: The destination object can be a partitioned table or a non-partitioned table. The partition key value cannot be in Chinese.

To upload the local file `banking.txt` to DataWorks, follow these steps:

1. On the **Data Analytics** tab, click the **Import** icon.
2. In the **Data Import Wizard** dialog box, select the table to which you want to import data and click **Next**.
3. Set **Select Data Import Method** to **Upload Local File** and click **Browse**. In the dialog box that appears, select the target local file and configure import information.

Parameter	Description
Select Data Import Method	The method of importing data. Valid values: Upload Local File , DataService Studio , and Workbooks from Data Analysis . In this example, select Upload Local File .
Select File	Click Browse and select the local file to upload.
Select Delimiter	The delimiter of fields in the file. Valid values: Comma , Tab , Semicolon , Space , , # , and & . In this example, select Comma .
Original Character Set	The character set of the file. Valid values: GBK , UTF-8 , CP936 , and ISO-8859 . In this example, select GBK .
Import First Row	The line from which data is to be imported. In this example, select 1 .
First Row as Field Names	Specifies whether to use the first line as the header line.
Data Preview	The preview of the data to import. Note If the data volume is large, only the data in the first 100 lines and 50 columns appears.

4. After the configuration is completed, click **Next**.
5. Select a matching mode for the fields in the source file and destination table. In this example, select **By Location**.
6. Click **Import Data**.

Data import methods

- Create a sync node

This method is used to import data from various data stores, such as Relational Database Service (RDS), MySQL, SQL Server, PostgreSQL, MaxCompute, ApsaraDB for Memcache, Distributed Relational Database Service (DRDS), Object Storage Service (OSS), Oracle, FTP, DM, Hadoop Distributed File System (HDFS), and MongoDB.

- Upload a local file

This method is used to upload .txt and .csv files not exceeding 10 MB. The destination object can be a partitioned table or a non-partitioned table. The partition key value cannot be in Chinese.

- Run Tunnel commands to upload a file

This method is used to upload local files and other resource files of any size.

What to do next

Now you have learned how to create tables and import data. You can proceed with the next tutorial. In the next tutorial, you will learn how to create a workflow and how to compute and analyze data in a workspace. For more information, see [Create a workflow](#).

2.3.3. Create a workflow

This topic describes how to create a workflow, create nodes in the workflow, and configure the dependencies among the nodes. After the configuration is completed, you can use the Data Analytics feature to further compute and analyze data in the workspace.

Prerequisites

The bank_data table for storing business data and the result_table table for storing data analytics results are created in the workspace. Data is imported to the bank_data table. For more information, see [Create tables and import data](#).

Context

The Data Analytics feature of DataWorks allows you to drag and drop nodes in a workflow and configure the dependencies among the nodes. You can process data and configure dependencies in the data based on the workflow.

Create a workflow

1. Log on to the DataWorks console.
2. On the **DataStudio** page that appears, move the pointer over the **Create** icon and click **Workflow**.
3. In the **Create Workflow** dialog box, set the **Business Name** and **Description** parameters.
4. Click **Create**.

Create nodes and configure dependencies among the nodes

This section describes how to create a zero load node named start and an ODPS SQL node named insert_data in the workflow, and configure the insert_data node to depend on the start node.

 **Note** Pay attention to the following points when you use a zero load node:

- A zero load node is a control node used to maintain and control its descendant nodes. When the zero load node runs in a workflow, it does not generate any data.
- If other nodes are dependent on the zero load node and it is manually set to Failed by an administration expert, the pending descendant nodes cannot be triggered. During the O&M process, an administration expert can disable the zero load node to prevent errors of ancestor nodes from being further expanded.
- Typically, the ancestor node of the zero load node in a workflow is set to the root node of the workspace. The root node of the workspace is named in the `Workspace name_root` format.

We recommend that you create a zero load node as the root node of a workflow to control the entire workflow.

1. Double-click the name of the workflow to go to the dashboard of the workflow. Move the pointer over **Zero-Load Node** and drag it to the development panel on the right.
2. In the **Create Node** dialog box, set **Node Name** to start and click **Commit**.
3. Repeat steps 1 and 2 to create an **ODPS SQL** node and name it `insert_data`.
4. Draw a line to connect the nodes and set the start node as the ancestor node of the `insert_data` node.

Configure the ancestor node of the zero load node

The zero load node in a workflow is the controller of the entire workflow, and also the ancestor of all nodes in the workflow. Generally, the zero load node in a workflow depends on the root node of the workspace.

1. Double-click the name of the zero load node. On the page that appears, click the **Properties** tab in the right-side navigation pane.
2. In the **Properties** section, click **Use Root Node** and set the ancestor node of the zero load node as the root node of the workspace.
3. After the configuration is completed, click  in the upper-left corner.

Edit code in the ODPS SQL node

This section provides a sample SQL statement used to query and save the number of singles with different education levels who loan to buy houses in the ODPS SQL node `insert_data`. The queried data can be analyzed by and presented in descendant nodes of `insert_data`.

The SQL statement is as follows:

```
INSERT OVERWRITE TABLE result_table --Insert data to the result_table table.
SELECT education
  , COUNT(marital) AS num
FROM bank_data
WHERE housing = 'yes'
  AND marital = 'single'
GROUP BY education
```

Run and debug the ODPS SQL node

1. After the SQL statement is entered in the `insert_data` node, click **Save**.
2. Click **Run** to view the runtime logs and result.

Commit the workflow

1. After running and debugging the ODPS SQL node `insert_data`, return to the workflow editing page and click **Commit**.
2. In the **Commit** dialog box, select the nodes to be committed, set **Description**, and then select **Ignore I/O Inconsistency Alerts**.
3. Click **Commit**.

What to do next

Now you have learned how to create and commit a workflow. You can proceed with the next tutorial. In the next tutorial, you will learn how to create a sync node to export data to different types of data stores. For more information, see [Create a sync node](#).

2.3.4. Create a sync node

This topic describes how to create a sync node to export data from MaxCompute to a MySQL database.

Background

In DataWorks, Data Integration can be used to periodically transfer the business data generated in a business system to a workspace. After the data is computed in SQL nodes, Data Integration periodically exports the computing results to your specified data store for further display or use.

Add a connection

 **Note** Only the workspace administrator can create connections, and members of other roles can only view the connections.

1. Log on to the DataWorks console. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration**.
2. In the left-side navigation pane, choose **Sync Resources > Connections**. On the **Connections** page, click **Create Connection**.
3. In the **Create Connection** dialog box, set **Data Source Type** to **MySQL**.
4. Set parameters in the **Add MySQL Connection** dialog box. The following table lists the parameters that need to be set when the connection type is set to **User-Created Data Store**.

Parameter	Description
Data Source Type	The type of the connection. In this example, set the type to MySQL > User-Created Data Store .
Data Source Name	The name of the connection. The name can contain letters, digits, and underscores (_) and must start with a letter.
Description	The description of the connection. The description cannot exceed 80 characters in length.
Applicable Environment	The environment where the connection is configured. Valid values: Development and Production .  Note This field is available only for workspaces in standard mode.

Parameter	Description
JDBC URL	The JDBC connectivity URL of the database, in the format of <code>jdbc:mysql://ServerIP:Port/Database</code> .
Username	The username for logging on to the database. <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px; margin-top: 10px;"> ? Note You must enter the information of your MySQL database. </div>
Password	The password for logging on to the database.

5. Click **Test Connection**.
6. If the connectivity test is successful, click **Complete**.

Verify that a table exists in the destination MySQL database

Use the following table creation statement to create the `odps_result` table in the MySQL database:

```
CREATE TABLE `ODPS_RESULT` (
  `education` varchar(255) NULL,
  `num` int(10) NULL
);
```

After the table is created, run the `desc odps_result;` statement to view the table details.

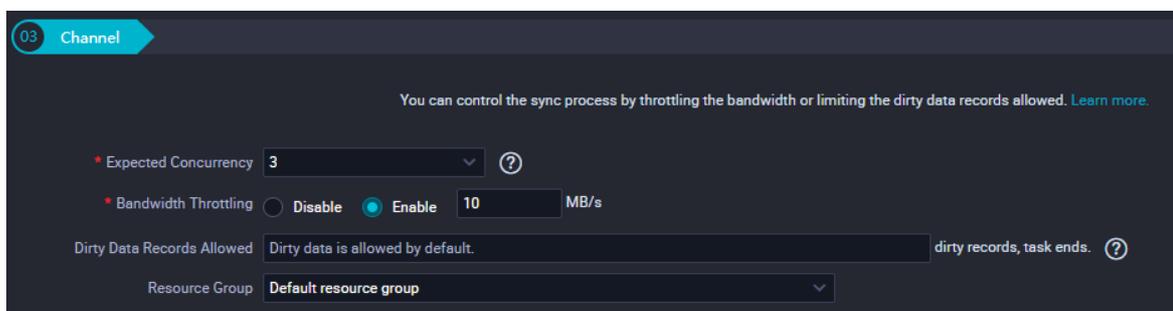
Create and configure a sync node

This section describes how to create and configure the sync node `write_result` to export data in the `result_table` table to your MySQL database. The procedure is as follows:

1. Go to the **Data Analytics** tab and create the sync node `write_result`.
2. Configure the `insert_data` node as the ancestor node of the `write_result` node.
3. Set **Data Source** to **MaxCompute > odps_first** and **Table** to `result_table`.
4. Select the `odps_result` table in your MySQL database as the destination table.
5. Configure the mapping between the fields in the source and destination tables.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add Line** to add a field or move the pointer over a field and click the **Delete** icon to delete the field.

6. In the Channel section, configure the synchronization rate limit and dirty data check rules.



Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The servers on which nodes are run. If an excessively large number of nodes are run in the default resource group, some nodes may be delayed due to insufficient resources. In this case, we recommend that you add a custom resource group.

7. Preview and save the configuration.

After the configuration is completed, scroll up and down to view the node configuration. Verify that the configuration is correct and click **Save**.

Commit the sync node

Return to the workflow after saving the sync node. In the top navigation bar, click **Commit** to commit the sync node to scheduling system. The scheduling system automatically and periodically runs the node starting from the next day based on the configured properties of the node.

What to do next

Now you have learned how to create a sync node to export data to a specific data store. You can proceed with the next tutorial. In the next tutorial, you will learn how to configure recurrence and dependencies for a sync node. For more information, see [Configure recurrence and dependencies for a node](#).

2.3.5. Configure recurrence and dependencies for a node

This topic describes how to configure recurrence and dependencies for a node in the DataWorks console.

 **Note** In this topic, the sync node `write_result` is used as an example and the recurrence is set to weekly.

DataWorks has a powerful scheduling engine to trigger nodes based on the recurrence and dependencies of nodes. DataWorks guarantees that tens of millions of nodes run accurately and punctually per day based on directed acyclic graphs (DAGs). In the DataWorks console, you can set the recurrence to minutely, hourly, daily, weekly, or monthly.

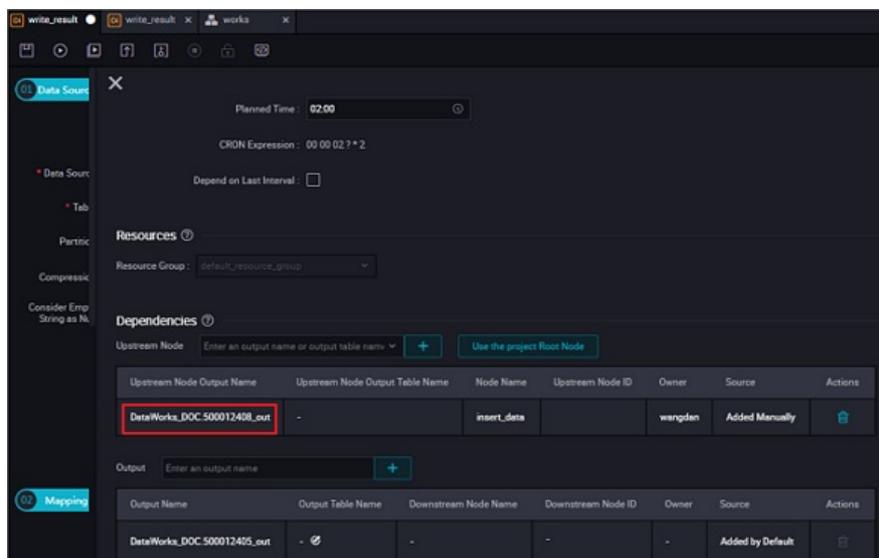
Configure recurrence for the sync node

1. After the sync node `write_result` is created, double-click the sync node to configure it.
2. Click the **Properties** tab in the right-side navigation pane to configure recurrence for the sync node.

Parameter	Description
Execution Mode	The mode in which the node is run. Valid values: Normal and Dry-Run . You can select one based on your own needs.
Retry Upon Error	Specifies whether to rerun the node upon an error.
Valid From	The date from which the node is effective.
Skip Execution	Specifies whether to skip execution of the node.
Cycle	The recurrence of the node, which can be monthly, weekly, daily, hourly, or minutely. In this example, the recurrence is set to weekly.
Customize Runtime	Specifies whether to run the node periodically. This field is selected by default.
Run Every or Run At	The specific day or time when the node is run. For example, you can configure a node to run at 02:00 every Tuesday.
CRON Expression	The value is <code>00 00 02 ? * 2</code> by default. It cannot be modified.
Cross-Cycle Dependencies	Specifies whether the node depends on the result of the last cycle.

Configure dependencies for the sync node

After configuring recurrence for the sync node `write_result`, you can continue to configure dependencies for the sync node.



You can configure the ancestor node on which the sync node depends. After that, the scheduling system can trigger the sync node when the specified time arrives, only after the instance of the ancestor node is run.

The configuration shown in the preceding figure indicates that the instance of the sync node is not triggered until the instance of the ancestor node `insert_data` is run.

The scheduling system creates the Workspace name_root node for each workspace as the root node by default. If no ancestor node is configured for the sync node, the sync node depends on the root node.

Commit the sync node

Save the configuration of the sync node `write_result` and click **Commit** to commit the node to the scheduling system.

Only after a node is committed, the scheduling system can automatically generate and run instances at the specified time starting from the next day according to the recurrence property.

 **Note** If a node is committed after 23:30, the scheduling system automatically generates and runs instances of the node starting from the third day.

What to do next

Now you have learned how to configure recurrence and dependencies for a sync node. You can proceed with the next tutorial. In the next tutorial, you will learn how to perform O&M on the committed node and troubleshoot errors based on the runtime logs. For more information, see [Run a node and troubleshoot errors](#).

2.3.6. Run a node and troubleshoot errors

This topic describes how to run and maintain a node, and troubleshoot errors based on logs.

When you configure recurrence and dependencies for the sync node `write_result`, you have configured the sync node to run at 02:00 every Tuesday. After you commit this node, you have to wait until the next day to view the automatic execution result of this node. DataWorks allows you to run nodes in the following modes: test run, retroactive run, and periodic run. This helps you confirm the run time of each node instance, dependencies among node instances, and whether generated data meets your expectation.

- **Test run:** Nodes are triggered manually. This method is recommended if you only want to confirm the run time and running of a single node.
- **Retroactive run:** Nodes are triggered manually. This method is recommended if you want to confirm the run time of multiple nodes and dependencies among them, or if you want to re-perform data analysis and computing from the specific root node.
- **Periodic run:** Nodes are triggered automatically. The scheduling system automatically triggers the instances of committed nodes at the specified time points starting from 00:00 the next day after the nodes are committed. In addition, the scheduling system checks whether the ancestor instances of each instance have been run when the scheduled time arrives. If all the ancestor instances have been run when the scheduled time arrives, the current instance is automatically triggered without manual intervention.

 **Note** The scheduling system generates instances for manually triggered nodes and auto triggered nodes based on the same rules.

- The scheduling system generates an instance for each recurrence, which can occur by day, hour, minute, month, or week.
- The scheduling system runs an instance only on the specified date and generates runtime logs for the instance.
- The scheduling system does not run an instance on other dates except the specified date. Instead, it directly change the status of the instance to successful when the running conditions are met. In this case, the scheduling system does not generate runtime logs.

Test run

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center** to go to the **Operation Center** page.
3. In the left-side navigation pane, click **Recurring**. On the page that appears, find the target node to run. Click **Test** next to the target node.

4. In the **Smoke Test** dialog box, set the **Smoke Test Instance Name** and **Data Timestamp** parameters and click **OK**.
5. On the **Smoke Test** page that appears, click an instance. The directed acyclic graph (DAG) of the instance appears on the right.

Right-click the instance to view its dependencies and details, and stop or re-run this instance.

 **Note**

- In test run mode, a node is triggered manually. The corresponding instance runs immediately when the scheduled time arrives, regardless of whether its ancestor instances have been run.
- The sync node `write_result` is configured to run at 02:00 every Tuesday. According to the instance generation rules described earlier in this topic, if the data timestamp, which is one day before the run date, is set to Monday for a test run, the scheduling system runs the instance for the sync node `write_result` at 02:00 on Tuesday. If the data timestamp is not set to Monday for the test run, the scheduling system changes the status of the instance to successful at 02:00 on Tuesday with no runtime logs generated.

Retroactive run

A retroactive run is recommended if you want to confirm the run time of multiple nodes and dependencies among them, or if you want to re-perform data analysis and computing from the specific root node.

1. On the **Operation Center** page, choose **Task List > Recurring** in the left-side navigation pane.
2. Find the target node to run and choose **Patch Data > Current Node Retroactively** for the target node.
3. In the **Patch Data** dialog box, set parameters and click **OK**.

Parameter	Description
Retroactive Instance Name	Enter the name of the retroactive instance.
Data Timestamp	Select the data timestamp of the retroactive instance. The retroactive instance is run on the next day of the specified timestamp.
Node	The default value is the current node, which cannot be changed.
Parallelism	Select Disable or specify several nodes to run concurrently.

4. On the **Retroactive** page that appears, click the retroactive instance to view the DAG of the instance. Right-click the instance to view its dependencies and details, and stop or re-run this instance.

 **Note**

- In retroactive run mode, instance running requires the result of instance running on the previous day. For example, retroactive instances are configured to run between September 15, 2017 and September 18, 2017. If the instance on September 15 fails to run, the instance on September 16 cannot run.
- The sync node `write_result` is configured to run at 02:00 every Tuesday. According to the instance generation rules described earlier in this topic, if the data timestamp, which is one day before the run date, is set to Monday for a retroactive instance, the scheduling system runs the instance for the sync node `write_result` at 02:00 on Tuesday. If the data timestamp is not set to Monday for the retroactive instance, the scheduling system changes the status of the instance to successful at 02:00 on Tuesday with no runtime logs generated.

Periodic run

In periodic run mode, the scheduling system automatically triggers instances for all nodes based on the scheduling configuration. No menu item is provided for you to control the periodic run on the DataStudio page. You can view the instance information and runtime logs in either of the following ways:

- On the **Operation Center** page, choose **Operation Center > Node O&M > Recurring** in the left-side navigation pane. On the page that appears, set parameters such as the data timestamp or run date, find an instance of the sync node `write_result`, and then right-click the instance to view the instance information and runtime logs.
- On the **Recurring** page, click the instance of the target node to view the DAG of the instance.

Right-click the instance to view its dependencies and details, and stop or re-run this instance.

 **Note**

- If an ancestor node is not run, a descendant node does not run either.
- If the initial status of an instance is pending, the scheduling system checks whether all its ancestor instances have been run when the scheduled time arrives.
- The instance can be triggered and run only after all its ancestor instances have been run and the scheduled time arrives.
- If an instance is pending, check whether all its ancestor instances have been run and whether the scheduled time arrives.

2.4. Data Integration

2.4.1. Overview

Data Integration is a stable, efficient, and scalable data synchronization service. It is designed to migrate and synchronize data between a wide range of heterogeneous data stores fast and stably in complex network environments.

Limits

- Data Integration can synchronize structured, semi-structured, and unstructured data. Structured data stores include Relational Database Service (RDS) and Distributed Relational Database Service (DRDS). Unstructured data, such as Object Storage Service (OSS) objects and text files, must be capable of being converted to structured data. Data Integration can only synchronize data that can be abstracted to two-dimensional logical tables to MaxCompute. It cannot synchronize unstructured data that cannot be converted to structured data, such as MP3 files stored in OSS, to MaxCompute.

- Data Integration supports data synchronization and exchange in one region or between regions.

Data can be transmitted between regions over the classic network, but the network connectivity is not guaranteed. If the transmission fails over the classic network, we recommend that you use an Internet connection.

- Data Integration supports only data synchronization but not data consumption.

Batch data synchronization

Data Integration can be used to synchronize large amounts of data. Data Integration facilitates data transmission between diverse structured and semi-structured data stores. It provides readers and writers for the supported data stores and defines a transmission channel between the source and destination data stores and datasets, based on simplified data types.

Supported data stores

- Relational databases: MySQL, SQL Server, PostgreSQL, Oracle, Dameng, DRDS, PolarDB, HybridDB for MySQL, AnalyticDB for PostgreSQL, AnalyticDB for MySQL 2.0, and AnalyticDB for MySQL 3.0
- Big data storage: MaxCompute, DataHub, and Data Lake Analytics (DLA)
- Semi-structured storage: OSS, Hadoop Distributed File System (HDFS), and FTP
- NoSQL: MongoDB, Memcache, Redis, and Tablestore
- Message queue: LogHub
- Graph compute engine: GraphCompute

For more information, see [Supported data sources](#).

 **Note** The connection configurations for data stores vary greatly. You can view the specific parameters that need to be set when you configure connections and sync nodes for data stores.

Development modes of sync nodes

You can develop sync nodes in one of the following modes:

- **Codeless UI:** Data Integration provides step-by-step instructions to help you configure a sync node. This mode is easy to use but provides only limited features.
- **Code editor:** You can write a JSON script to create a sync node. This mode supports advanced features to facilitate flexible configuration. It is suitable for experienced users and increases the cost of learning.

 **Note**

- The code generated for a sync node on the codeless user interface (UI) can be converted to a script. This conversion is irreversible.
- Before you write code, you must configure a connection and create the destination table.

Network types

A data store can reside on the classic network or in a virtual private cloud (VPC). The user-created IDC network type has been planned and will be supported soon.

- **Classic network:** a network deployed by Alibaba Cloud, which is shared with other tenants. This network is easy to use.
- **VPC:** a network created on Alibaba Cloud, which is available to only one Apsara Stack account. You have full control over your VPC, including customizing the IP address range, dividing the VPC to multiple subnets, and configuring routing tables and gateways.

A VPC is an isolated network for which you can customize a wide range of parameters, such as the IP address range, subnets, and gateways. Based on wide deployment of VPCs, Data Integration provides the feature to automatically detect the reverse proxy for some data stores, including ApsaraDB RDS for MySQL, ApsaraDB RDS for PostgreSQL, ApsaraDB RDS for SQL Server, PolarDB, DRDS, HybridDB for MySQL, AnalyticDB for PostgreSQL, and AnalyticDB for MySQL 3.0. By using this feature, you do not need to purchase an extra Elastic Compute Service (ECS) instance in your VPC to configure sync nodes for these data stores. Instead, Data Integration automatically uses this feature to provide network connectivity to these data stores.

When you configure sync nodes for other Alibaba Cloud data stores in a VPC, such as PPAS, ApsaraDB for OceanBase, ApsaraDB for Redis, ApsaraDB for MongoDB, ApsaraDB for Memcache, Tablestore, and ApsaraDB for HBase, you must purchase an ECS instance in the same VPC. This ECS instance is used to access the data stores.

- **User-created IDC network:** an IDC network deployed by yourself, which is isolated from the Alibaba Cloud network.

 **Note** You can access data stores over the Internet. However, the access speed depends on the Internet bandwidth, and additional network access expenses are required. We recommend that you do not use Internet connections.

Terms

- **Concurrency**

Concurrency indicates the maximum number of concurrent threads to read data from or write data to data storage within a single sync node.

- **Bandwidth throttling**

Bandwidth throttling indicates that a maximum transmission rate is specified for a sync node of Data Integration.

- **Dirty data**

Dirty data indicates meaningless data and data that does not match the specified data type. For example, you want to write data of the VARCHAR type in the source table to an INT-type field in the destination table. A data conversion error occurs and the data cannot be written to the destination table. In this case, the data is dirty.

- **Connection**

A connection in DataWorks is used for accessing a data store, which can be a database or a data warehouse. DataWorks supports various types of data stores, and supports data synchronization between data stores of different types.

2.4.2. Homepage

The Data Integration homepage provides entries for you to create sync nodes, manage connections, maintain sync nodes, and view help documents.

Log on to the [DataWorks console](#), click  in the upper-left corner, and choose **All Products > Data Aggregation > Data Integration**. The homepage of Data Integration appears by default.

On this page, you can perform the following operations:

- **New Task:** Click here to go to the **Data Analytics** page, where you can create sync nodes. For more information, see [Create a sync node](#).
- **Connection:** Click here to go to the **Data Source** page, where you can view created connections and add a connection or multiple connections at a time.

- **Workbench:** Click here to go to the **Operation Center > Dashboard** page, where you can view the running status of created nodes. For more information, see [Dashboard](#).

2.4.3. Connectivity testing

This topic describes the FAQ about connectivity testing on connections.

When configuring a security group for a connection hosted on an Elastic Compute Service (ECS) instance, add the IP address of the scheduling cluster to the inbound and outbound rules of the security group. If the security group is not properly configured, data synchronization fails due to a connection failure.

To set a wide port range for a security group rule, call relevant API operations, instead of using the console.

Common scenarios of connectivity test failures

When a connection fails the connectivity test, check whether the region, network type, whitelist, database name, and username are properly configured for the connection. The following errors may occur during connectivity testing:

- The database password is incorrect.
- The network connection fails.
- A network error occurs during data synchronization.

Check the log and determine which resource group is used. Check whether the resource group is a custom one.

For a Relational Database Service (RDS) connection or a MongoDB connection, if a custom resource group is used, check whether its IP addresses are added to the whitelist of the connection.

Check whether both the source and destination connections pass the connectivity test. For an RDS connection or a MongoDB connection, check whether all relevant IP addresses are added to the whitelist of the connection. If the IP address of a server is not added to the whitelist, the sync node fails when it runs on this server. However, the sync node succeeds when it runs on another server whose IP address is added to the whitelist.

- The result shows that a sync node is run but the log contains a disconnection error in port 8000.

This issue occurs because a custom resource group is used and no inbound rule is configured for the corresponding IP address and port 8000 in the security group. To resolve the issue, add the IP address and port to the inbound rule of the security group and run the node again.

Examples of connectivity test failures

Example 1

- Symptom

A connection failed the connectivity test. The database connection failed. The following information is involved: Database URL: jdbc:mysql://xx.xx.xx.x:xxxx/t_uoer_bradeF. Username: xxxx_test. Error message: Access denied for user 'xxxx_test'@'%' to database 'yyyy_demo'.

- Troubleshooting

- i. Check whether the configuration of the connection is correct.
- ii. Check whether the database password is correct, the whitelist is properly configured, and your account has the permission to access the database. You can grant the required permissions in the RDS console.

- Example 2

- Symptom

A connection failed the connectivity test. The following error message is returned:

```
error message: Timed out after 5000 ms while waiting for a server that matches ReadPreferenceServerSelector{readPreference=primary}. Client view of cluster state is {type=UNKNOWN, servers=[(xxxxxxxx), type=UNKNOWN, state=CONNECTING, exception={com.mongodb.MongoSocketReadException: Prematurely reached end of stream}]}
```

- Troubleshooting

Before testing the connectivity to a MongoDB connection that is not deployed in a Virtual Private Cloud (VPC), add relevant IP addresses to the whitelist of the connection.

2.4.4. Data sources

2.4.4.1. Supported data stores and plug-ins

Data Integration is a stable, efficient, and scalable data synchronization service. It provides transmission channels for batch data stored in Alibaba Cloud services such as MaxCompute, AnalyticDB for PostgreSQL, and Hologres.

The following table lists the data stores and plug-ins that Data Integration supports.

Data store	Reader	Writer
ApsaraDB for OceanBase	ApsaraDB for OceanBase	ApsaraDB for OceanBase
DataHub	DataHub Reader	DataHub Writer
Db2	DB2 Reader	DB2 Writer
DM	RDBMS Reader	RDBMS Writer
DRDS	DRDS Reader	DRDS Writer
Elasticsearch	Elasticsearch Reader	Elasticsearch Writer
FTP	FTP	FTP Writer
GBase8a	Supported	GBase8a Writer
HBase	HBase Reader	<ul style="list-style-type: none"> • HBase Writer • HBase11xsql Writer
HDFS	HDFS Reader	HDFS Writer
Hive	Hive Reader	Hive Writer
Hologres	Supported	Supported
HybridDB for MySQL	Supported	Supported
LogHub	LogHub Reader	LogHub Writer
MaxCompute	MaxCompute Reader	MaxCompute Writer

Data store	Reader	Writer
Memcache	Not supported	Memcache Writer
MongoDB	MongoDB Reader	MongoDB Writer
MySQL	MySQL Reader	MySQL Writer
Oracle	Oracle Reader	Oracle Writer
OSS	OSS Reader	OSS Writer
POLARDB	Supported	Supported
PostgreSQL	PostgreSQL Reader	PostgreSQL Writer
RDBMS	RDBMS Reader	RDBMS Writer
Redis	Not supported	Redis Writer
Stream	Stream Reader	Stream Writer
SQL Server	SQL Server Reader	SQL Server Writer
Tablestore	Tablestore Reader	Tablestore Writer
Vertica	Vertica Reader	Vertica Writer

2.4.4.2. Connection isolation

DataWorks provides the connection isolation feature to isolate data of the development environment from that of the production environment for workspaces in standard mode.

If a connection is configured in both the development and production environments, you can use the connection isolation feature to isolate the connection in the development environment from that in the production environment.

 **Note** Currently, only workspaces in standard mode support the connection isolation feature.

When you configure a sync node, the connection in the development environment is used. After you commit and deploy the sync node to the production environment for running, the connection in the production environment is used. To commit and deploy a node to the production environment for scheduling, you must configure a connection in both the development and production environments. The connection must have the same name in the development and production environments.

The connection isolation feature has the following impacts on workspaces:

- Workspaces in basic mode: The features and configuration dialog boxes of connections are the same as those before the connection isolation feature is added. For more information, see [Connection configuration](#).
- Workspaces in standard mode: The Applicable Environment parameter is added to the configuration dialog boxes of connections.
- Workspaces upgraded from the basic mode to the standard mode: During the upgrade, you are prompted to upgrade connections. After the upgrade, the connections in the development environment are isolated from those in the production environment.

2.4.4.3. Sync data monitoring

The Sync Data Monitoring page displays the total number of sync node instances for different connections and the instance details based on the selected workspace and time range.

The cut-off time of data to be displayed is 0 minutes 0 seconds of the current hour. For example, if the current time is 2019-04-04 10:10:00, the page displays the data generated before 2019-04-04 10:00:00.

1. Log on to the DataWorks console and select a workspace.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page.
3. In the left-side navigation pane, click **Sync Data Monitoring**. On the page that appears, view the total number of sync node instances for different connections and the instance details.

- o View summary data by connection type

The Source and Target sections display the summary data of source connections and that of destination connections, respectively. Take source connections as an example. If the Source section displays MaxCompute with the value 1, a sync node instance whose source connection is MaxCompute is run in the selected time range.

- o View instance details

The Sync Instances section displays the details of all sync node instances that are run in the selected time range. You can also perform the following operations:

- Click a node in the **Node Name** column to go to the node configuration page.
- Search for instances by condition, such as the ID, committer, node name, source connection type, and destination connection type. Sort search results based on the number of synchronized data entries or the size of synchronized data.

2.4.4.4. Manage connection permissions

DataWorks allows you to share connections among workspaces by managing permissions on the connections. After connections are shared, you can view the shared connections in the target workspaces. This topic describes how to manage permissions on connections and view shared connections.

Context

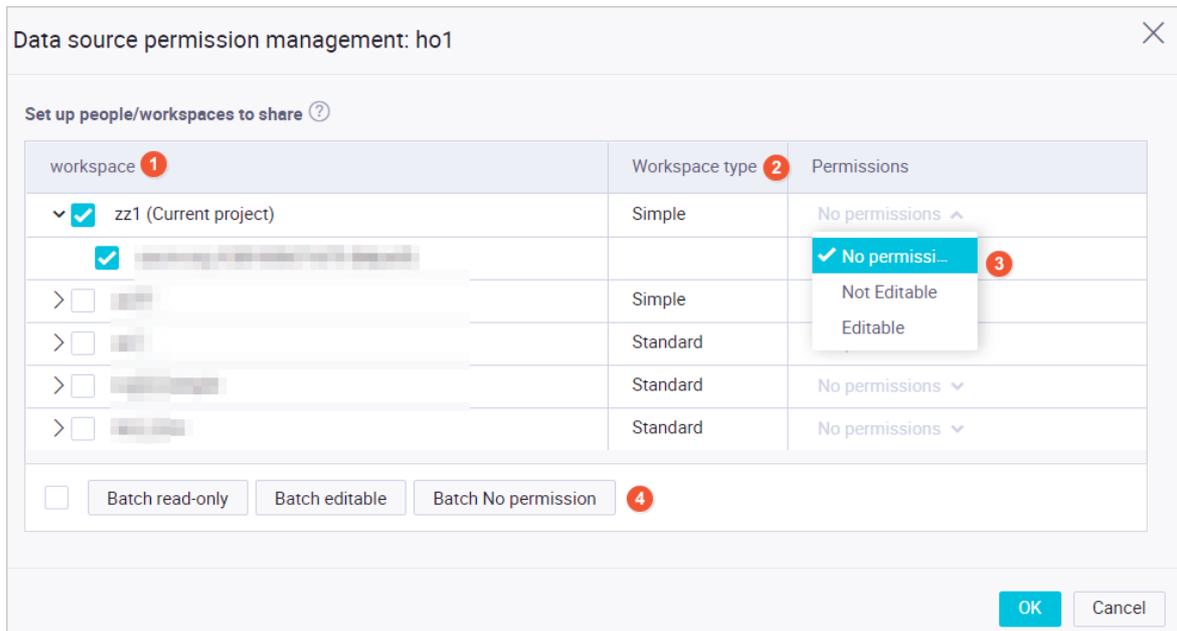
The configurations of a connection include sensitive information such as the endpoint of the data store, username, and password. Common developers only need to reference the connection to access the data store. Disclosing too much sensitive information or allowing everyone to modify the configurations of the connection may cause security risks. If multiple users modify the configurations of a connection, the data store may fail to be connected. In this way, the nodes that reference the connection may fail.

Data Integration provides strict permission control. Only connection creators can manage the permissions on connections. They can grant permissions on connections to a specified workspace or user.

Go to the Data Source page

1. Log on to the DataWorks console.
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > Data Integration**.
3. On the page that appears, click **Data Source** in the left-side navigation pane.
4. On the page that appears, find the target connection and click **Modify Permission** in the **Actions** column.
5. In the **Data source permission management** dialog box, set the parameters as described in the

following table.



No.	Parameter	Description
1	Workspace	<p>All workspaces that the current user joins and all members in each workspace. You can share the connection with several or all members in a workspace.</p> <ul style="list-style-type: none"> ◦ If no permission is set for a connection, the connection inherits the permissions from the connection that is created earlier than the current one. ◦ When you configure the permissions on a connection for a workspace, the permissions apply to all members in the workspace. Members that join the workspace after the permission configuration also have the specified permissions. After you configure the permissions for a workspace, you can still configure the permissions for a specific user in the workspace. For example, after you set the permission on a connection to No permission for a workspace, you can still set the permission of a specific user in the workspace to Editable. ◦ You can configure the permissions on a connection for members in the current workspace. ◦ Only the creator of a connection can modify and share the connection. Other users including the workspace administrator cannot modify the connection. ◦ A workspace administrator can use a connection only after the workspace administrator is granted the required permission.
2	Workspace type	The type of each workspace. Valid values: Simple and Standard .

No.	Parameter	Description
3	Permissions	<p>The permission of a workspace or a member on the connection. Valid values:</p> <ul style="list-style-type: none"> ◦ No permission: The workspace or member has no permission on the connection. ◦ Not Editable: The workspace or member can use the connection but cannot modify or view the configurations of connection. ◦ Editable: The workspace or member can use and modify the connection. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note If you grant the Editable permission with a workspace or member, the workspace or member can modify the connection. Exercise caution when you grant the Editable permission.</p> </div>
4	Batch operations	<p>The operations that you can perform on the selected workspaces or members at a time. Valid values: Batch read-only, Batch editable, and Batch No permission.</p>

6. Click **OK**. You can share connections across workspaces based on the following rules:

- Between workspaces in simple mode:
 - If the source workspace is upgraded to the standard mode, connections in the production environment are shared.
 - If the target workspace is upgraded to the standard mode, a connection is shared to both the development environment and production environment with the same content.
- From a workspace in simple mode to a workspace in standard mode: A connection is shared to both the development environment and production environment with the same content.
- Between workspaces in standard mode: Connections in the development environment and production environment are shared to the corresponding environment separately.
- From a workspace in standard mode to a workspace in simple mode:
 - You can share connections in both the production environment and development environment. Only connections in the production environment or development environment exist in the target workspace. If you share a connection in both environments, the newly shared one overrides the existing one in the target workspace.
 - If the target workspace is upgraded to the standard mode, the shared connection exists in both the development environment and production environment with the same content.

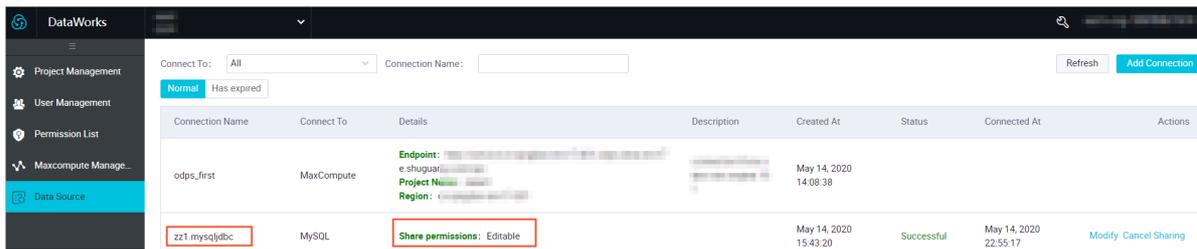
View shared connections

In the top navigation bar, select a workspace with connections shared from other workspaces from the drop-down list in the upper-left corner. The **Data Source** page of the selected workspace appears. On this page, you can view shared connections on the **Normal** and **Has expired** tabs.

• Normal tab

On the Normal tab, you can view the information about each connection, including the connection name, connection type, permission details, connection description, creation time, connection status, and the time when the data store was last connected.

The permission information appears in the **Details** column of the target connection. A shared connection is named in the Name of the workspace that shares the connection. Connection name format.

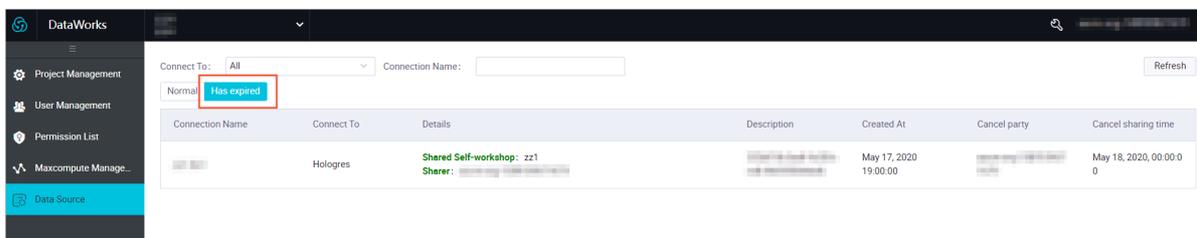


If the current user has the Editable permission on the connection, **Modify** appears in the Actions column.

- **Has expired tab**

On the **Has expired** tab, you can view the connections for which your permissions have expired.

In the **Cancel party** column, you can view the member who revoked the permissions. In the **Created at** column, you can view the time when the permissions were revoked. The information helps you locate the cause of connection failures.



2.4.4.5. Configure a MySQL connection

A MySQL connection allows you to read data from and write data to MySQL by using MySQL Reader and Writer. You can configure sync nodes for MySQL by using the codeless UI or code editor.

Procedure

- Go to the **Data Source** page.
 - Log on to the **DataWorks console**.
 - On the **DataStudio** page, click  in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
- Click **Add Connection** in the upper-right corner.
- In the **Add Connection** dialog box, click **MySQL** in the **Relational Databases** section.
- In the **Add MySQL Connection** dialog box, set the parameters as required. You can set **Connect To** parameter to **ApsaraDB for RDS** or **Connection Mode** for a MySQL connection.
 - The following table describes the parameters that appear after you set the **Connect To** parameter to **ApsaraDB for RDS**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for RDS .

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
RDS Instance ID	The ID of the ApsaraDB RDS for MySQL instance. You can view the ID in the ApsaraDB for RDS console.
RDS Instance Account ID	The ID of the Apsara Stack tenant account that is used to purchase the ApsaraDB RDS for MySQL instance.
Database Name	The name of the database.
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

- o The following table describes the parameters that appear after you set the **Connect To** parameter to **Connection Mode**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:mysql://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

Parameter	Description
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test Connection**.
6. After the connection passes the connectivity test, click **Complete**.

What's next

Now you have learned how to configure a MySQL connection. You can proceed with the next tutorial. In the next tutorial, you will learn how to configure MySQL Reader and Writer. For more information, see [Configure the MySQL reader](#) and [Configure MySQL Writer](#).

2.4.4.6. Configure an SQL Server data source

DataWorks provides SQL Server Reader and SQL Server Writer for you to read data from and write data to SQL Server data sources. You can use the codeless user interface (UI) or code editor to configure synchronization nodes for SQL Server data sources.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add data source** dialog box, click **SQLServer** in the Relational Databases section.
4. In the **Add SQLServer data source** dialog box, configure the parameters. You can set the Data source type parameter to **Alibaba Cloud instance mode** or **Connection string mode** for an SQL Server data source.
 - o The following table describes the parameters that appear after you set **Data source type** to **Alibaba Cloud instance mode**.

Parameter	Description
Data source type	The type of the data source. Set the parameter to Alibaba Cloud instance mode .
Data Source Name	The name of the data source. The name must contain letters, digits, and underscores (_) and start with a letter.
Description	The description of the data source. The description can be a maximum of 80 characters in length.

Parameter	Description
Environment	<p>The environment in which the data source is used. Valid values: Development and Production.</p> <p> Note This parameter is displayed only when the workspace is in standard mode.</p>
RDS instance ID	The ID of the ApsaraDB RDS for SQL Server instance. You can view the ID in the ApsaraDB RDS console.
RDS instance account ID	The ID of the Apsara Stack tenant account that is used to purchase the ApsaraDB RDS for SQL Server instance.
Database name	The name of the ApsaraDB RDS for SQL Server database.
User name	The username that is used to connect to the database.
Password	The password that is used to connect to the database.

- o The following table describes the parameters that appear after you set **Data source type** to **Connection string mode**.

Parameter	Description
Data source type	The type of the data source. Set the parameter to Connection string mode .
Data Source Name	The name of the data source. The name must contain letters, digits, and underscores (_) and start with a letter.
Description	The description of the data source. The description can be a maximum of 80 characters in length.
Environment	<p>The environment in which the data source is used. Valid values: Development and Production.</p> <p> Note This parameter is displayed only when the workspace is in standard mode.</p>
JDBC URL	The Java Database Connectivity (JDBC) URL of the database. In the format of <code>jdbc:sqlserver://ServerIP:Port;DatabaseName=Database</code> .
User name	The username that is used to connect to the database.
Password	The password that is used to connect to the database.
Whether the data source is in a VPC	Specifies whether to connect to the data source by using a VPC. If you cannot connect to the ECS instance where the data source is located but can connect to the VPC to which the data source belongs, select the check box.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

What's next

You have learned how to configure an SQL Server data source. You can proceed to subsequent tutorials. The subsequent tutorials describe how to configure SQL Server Reader or SQL Server Writer. For more information, see [Configure SQL Server Reader](#) or [Configure SQL Server Writer](#).

2.4.4.7. Configure a PostgreSQL connection

A PostgreSQL connection allows you to read data from and write data to PostgreSQL by using PostgreSQL Reader and Writer. You can configure sync nodes for PostgreSQL by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **PostgreSQL** in the Relational Databases section.
4. In the **Add PostgreSQL Connection** dialog box, set the required parameters. You can set the **Connect To** parameter to **ApsaraDB for RDS** or **Connection Mode** for a PostgreSQL connection.
 - o The following table describes the parameters that appear after you set the **Connect To** parameter to **ApsaraDB for RDS**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for RDS .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
RDS Instance ID	The ID of the ApsaraDB RDS for PostgreSQL instance. You can view the ID in the ApsaraDB for RDS console.
RDS Instance Account ID	The ID of the Apsara Stack tenant account that is used to purchase the ApsaraDB RDS for PostgreSQL instance. You can view your account ID on the Security Settings page.
Database Name	The name of the database.

Parameter	Description
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

- o The following table describes the parameters that appear after you set the **Connect To** parameter to **Connection Mode**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e0f0ff;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:postgresql://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.8. Configure an Oracle connection

An Oracle connection allows you to read data from and write data to Oracle by using Oracle Reader and Writer. You can configure sync nodes for Oracle by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.

2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Oracle** in the Relational Databases section.
4. In the **Add Oracle Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	<p>The environment in which the connection is used. Valid values: Development and Production.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:oracle:thin:@ServerIP:Port:Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.9. Configure a Dameng connection

A Dameng connection allows you to read data from and write data to Dameng by using Dameng Reader and Writer. You can configure sync nodes for Dameng by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **DM** in the Relational Databases section.
4. In the **Add DM Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:dm://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.10. Configure a DRDS connection

A DRDS connection allows you to read data from and write data to DRDS by using DRDS Reader and Writer. You can configure sync nodes for DRDS by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **DRDS** in the Relational Databases section.
4. In the **Add DRDS Connection** dialog box, set the parameters as required. You can set the **Connect To** parameter to **ApsaraDB for DRDS** or **Connection Mode** for a DRDS connection.
 - o The following table describes the parameters that appear after you set the **Connect To** parameter to **ApsaraDB for DRDS**.

Parameter	Description
-----------	-------------

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for DRDS .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Instance ID	The ID of the DRDS instance. You can view the ID in the DRDS console.
Tenant Account ID	The ID of the Apsara Stack tenant account that is used to purchase the DRDS instance. You can view your account ID on the Security Settings page.
Database Name	The name of the database.
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

- o The following table describes the parameters that appear after you set the **Connect To** parameter to **Connection Mode**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:mysql://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.

Parameter	Description
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

What's next

Now you have learned how to configure a DRDS connection. You can proceed with the next tutorial. In the next tutorial, you will learn how to configure DRDS Reader and Writer. For more information, see [Configure the DRDS reader](#).

2.4.4.11. Configure a PolarDB connection

A PolarDB connection allows you to read data from and write data to PolarDB by using PolarDB Reader and Writer. You can configure sync nodes for PolarDB by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **POLARDB** in the Relational Databases section.
4. In the **Add POLARDB Connection** dialog box, set the parameters as required.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
Database Type	The type of the database. Valid values: MySQL and Postgresql .
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:mysql://ServerIP:Port/Database</code> .

Parameter	Description
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.12. Configure a HybridDB for MySQL connection

A HybridDB for MySQL connection allows you to read data from and write data to HybridDB for MySQL by using HybridDB for MySQL Reader and Writer. You can configure sync nodes for HybridDB for MySQL by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **HybridDB for MySQL** in the Relational Databases section.
4. In the **Add HybridDB for MySQL Connection** dialog box, set the parameters as required.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for AnalyticDB .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
Instance ID	The ID of the HybridDB for MySQL instance. You can view the ID in the HybridDB for MySQL console.
Tenant Account ID	The ID of the Apsara Stack tenant account that is used to purchase the HybridDB for MySQL instance.
Username	The username that you can use to connect to the database.

Parameter	Description
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.13. Configure a HybridDB for PostgreSQL connection

A HybridDB for PostgreSQL connection allows you to read data from and write data to HybridDB for PostgreSQL by using HybridDB for PostgreSQL Reader and Writer. You can configure sync nodes for HybridDB for PostgreSQL by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **HybridDB for PostgreSQL** in the Relational Databases section.
4. In the **Add HybridDB for PostgreSQL Connection** dialog box, set the parameters as required.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for AnalyticDB .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
Instance ID	The ID of the HybridDB for PostgreSQL instance. You can view the ID in the HybridDB for PostgreSQL console.
Tenant Account ID	The ID of the Apsara Stack tenant account that is used to purchase the HybridDB for PostgreSQL instance. You can view your account ID on the Security Settings page.
Database Name	The name of the database.

Parameter	Description
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.14. Configure an ApsaraDB for OceanBase connection

An ApsaraDB for OceanBase connection allows you to read data from and write data to ApsaraDB for OceanBase by using ApsaraDB for OceanBase Reader and Writer. You can configure sync nodes for ApsaraDB for OceanBase by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **ApsaraDB for OceanBase** in the Big Data Storage Systems section.
4. In the **Add ApsaraDB for OceanBase Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
JDBC URL	The JDBC URL of the ApsaraDB for OceanBase database, in the format <code>jdbc: oceanbase://ip:port/database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.15. Configure a MaxCompute connection

A MaxCompute connection allows you to read data from and write data to MaxCompute by using MaxCompute Reader and Writer.

Context

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **MaxCompute** in the Big Data Storage Systems section.
4. In the **Add MaxCompute Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 5px;">? Note This parameter is available only when the workspace is in standard mode.</div>
ODPS Endpoint	The endpoint of the MaxCompute project. This parameter is read-only, and the value is automatically obtained from system configurations.
Tunnel Endpoint	The endpoint of the MaxCompute Tunnel service.
MaxCompute Project Name	The name of the MaxCompute project.
AccessKey ID	The AccessKey ID for connecting to the MaxCompute project.
AccessKey Secret	The AccessKey secret for connecting to the MaxCompute project.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.16. Configure a DataHub connection

DataHub offers a comprehensive data import scheme to support fast computing for large amounts of data.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console.](#)
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **DataHub** in the Big Data Storage Systems section.
4. In the **Add DataHub Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
DataHub Endpoint	The endpoint of DataHub. This parameter is read-only, and the value is automatically obtained from system configurations.
DataHub Project	The ID of the DataHub project.
AccessKey ID	The AccessKey ID for connecting to the DataHub project. You can view the AccessKey ID on the User Info page.
AccessKey Secret	The AccessKey secret for connecting to the DataHub project.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.17. Configure an AnalyticDB for MySQL connection

An AnalyticDB for MySQL connection allows you to write data to AnalyticDB for MySQL by using AnalyticDB for MySQL Writer. You can configure sync nodes for AnalyticDB for MySQL by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console.](#)
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.

- iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **ADS** in the Big Data Storage Systems section.
4. In the **Add ADS Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
Connection URL	The connection URL of AnalyticDB for MySQL, in the format of <code>Address:Port</code> .
Database	The name of the database.
AccessKey ID	The AccessKey ID for connecting to the AnalyticDB for MySQL database.
AccessKey Secret	The AccessKey secret for connecting to the AnalyticDB for MySQL database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.18. Configure a Vertica connection

A Vertica connection allows you to read data from and write data to Vertica by using Vertica Reader and Writer. You can configure sync nodes for Vertica by using the UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Vertica** in the Big Data Storage Systems section.
4. In the **Add Vertica Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
JDBC URL	The JDBC URL of the Vertica database, in the format of <code>jdbc:vertica://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.19. Configure a GBase8a connection

A GBase8a connection allows you to read data from and write data to GBase8a by using GBase8a Reader and Writer. You can configure sync nodes for GBase8a by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **GBase8a** in the Big Data Storage Systems section.
4. In the **Add GBase8a Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.

Parameter	Description
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	<p>The environment in which the connection is used. Valid values: Development and Production.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
JDBC URL	The JDBC URL of the database, in the format of <code>jdbc:mysql://ServerIP:Port/Database</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.20. Configure a Lightning connection

MaxCompute Lightning is an interactive query service that MaxCompute provides. MaxCompute Lightning complies with the PostgreSQL standards and syntax and allows you to use common tools and standard SQL to query and analyze data in MaxCompute projects.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Lightning** in the Big Data Storage Systems section.
4. In the **Add Lightning Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.

Parameter	Description
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Host	The endpoint of the MaxCompute Lightning server. Default value: <code>sea-hawks.aliyun-inc.com</code> .
Port	The port number of the MaxCompute Lightning server. Default value: 8099.
Database Name	The name of the database.
Username and Password	The username and password that you can use to connect to the database.
ODPS Endpoint	The endpoint of MaxCompute.
MaxCompute Project Name	The name of the MaxCompute project.
AccessKey ID	The AccessKey ID for connecting to the MaxCompute Lightning server.
AccessKey Secret	The AccessKey secret for connecting to the MaxCompute Lightning server.
JDBC Extension Parameters	The extension parameters used to establish a JDBC connection to MaxCompute Lightning. In this field, <code>prepareThreshold=0</code> is added by default and cannot be deleted. Otherwise, you cannot connect to MaxCompute Lightning.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.21. Configure an HBase connection

An HBase connection allows you to read data from and write data to HBase by using HBase Reader and Writer. You can configure sync nodes for HBase by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **HBase** in the Big Data Storage Systems section.
4. In the **Add HBase Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	<p>The environment in which the connection is used. Valid values: Development and Production.</p> <div style="background-color: #e0f2f1; padding: 5px; border: 1px solid #ccc;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
Configuration	<p>The HBase cluster configuration for client connections.</p> <p>You can convert the hbase-site.xml parameter to the JSON format and add more HBase client properties, such as cache and batch for scan operations, to optimize the interaction between the cluster and the client.</p> <p>Based on the edition of ApsaraDB for HBase in use, you must configure different information:</p> <ul style="list-style-type: none"> ◦ If you are using ApsaraDB for HBase Standard Edition or less advanced editions, the default configuration is used. You only need to enter the corresponding ZooKeeper information. ◦ If you are using ApsaraDB for HBase editions that are more advanced than Standard Edition, the endpoint parameter specific to advanced editions is used for connection, and the zookeeper.quorum parameter is not used. <p>The following configuration is an example for an HBase connection of ApsaraDB for HBase Enhanced Edition (Lindorm):</p> <pre style="background-color: #f5f5f5; padding: 10px; border: 1px solid #ccc;"> "hbaseConfig": { "hbase.client.connection.impl" : "com.alibaba.hbase.client.AliHBaseUEConnection", "hbase.client.endpoint" : "host:30020", "hbase.client.username" : "root", "hbase.client.password" : "root" }</pre>

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.22. Configure a Hologres connection

A Hologres connection allows you to read data from and write data to Hologres by using Hologres Reader and Writer. You can configure sync nodes for Hologres by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.

- iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Hologres** in the Big Data Storage Systems section.
4. In the **Add Hologres Connection** dialog box, set the parameters as required.

Parameter	Description
Connect To	The type of the connection. Default value: ApsaraDB for RDS .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
Instance ID	The ID of the Hologres instance.
Database Name	The name of the database in the Hologres instance.
AccessKey ID	The AccessKey ID for connecting to the Hologres database.
AccessKey Secret	The AccessKey secret for connecting to the Hologres database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.23. Configure a Hive connection

A Hive connection allows you to read data from and write data to Hive by using Hive Reader and Writer. You can configure sync nodes for Hive by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Hive** in the Big Data Storage Systems section.
4. In the **Add Hive Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	<p>The environment in which the connection is used. Valid values: Development and Production.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
HIVE JDBC URL	The JDBC URL of the Hive metadatabase.
Database Name	The name of the Hive database. You can run the <code>show databases</code> command on the Hive client to query the created databases.
Logon Method	<p>The mode for connecting to the Hive database. Valid values: Username and password and Anonymous logon.</p> <p>If you select Username and password, enter the username and password that you can use to connect to the Hive database.</p>
metastoreUri	The URIs of the Hive metadatabase, in the format of <code>thrift://ip1:port1,thrift://ip2:port2</code> .
defaultFS	The address of the NameNode in the Active state in the HDFS, in the format of <code>hdfs://ip:port</code> .
Extension Parameters	<p>The advanced parameters of Hive, such as those related to high availability (HA). The following code is an example:</p> <pre>"hadoopConfig":{ "dfs.nameservices": "testDfs", "dfs.ha.namenodes.testDfs": "namenode1,namenode2", "dfs.namenode.rpc-address.youkuDfs.namenode1": "", "dfs.namenode.rpc-address.youkuDfs.namenode2": "", "dfs.client.failover.proxy.provider.testDfs": "org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider" }</pre>

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.24. Configure an OSS connection

Alibaba Cloud OSS is a secure and reliable service that allows you to store large amounts of objects.

Procedure

1. Go to the **Data Source** page.

- i. Log on to the DataWorks console.
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
 3. In the **Add Connection** dialog box, click **OSS** in the Semi-Structured Storage Systems section.
 4. In the **Add OSS Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Endpoint	The OSS endpoint, in the format of <code>http://oss.aliyuncs.com</code> . The OSS endpoint varies with the region.  Note If you add the bucket name before the domain name, for example, <code>http://xxx.oss.aliyuncs.com</code> , the connection can pass the connectivity test but data synchronization will fail.
Bucket	The name of the OSS bucket. A bucket is a storage space that serves as a container for storing objects. You can create one or more buckets and add one or more objects to each bucket. DataWorks can search for objects only in the bucket specified here during data synchronization.
AccessKey ID	The AccessKey ID for connecting to the OSS bucket.
AccessKey Secret	The AccessKey secret for connecting to the OSS bucket.

 **Notice** When data in OSS is stored as CSV files, they must comply with the standard CSV format. For example, if the data in a column of a CSV file contains a double quotation mark ("), you must replace the double quotation mark with a pair of double quotation marks ("""). Otherwise, the data in the CSV file may be incorrectly parsed.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.25. Configure an HDFS connection

A HDFS connection allows you to read data from and write data to HDFS by using HDFS Reader and Writer. You can configure sync nodes for HDFS by using the code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **HDFS** in the Semi-Structured Storage Systems section.
4. In the **Add HDFS Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> Note This parameter is available only when the workspace is in standard mode.</div>
DefaultFS	The address of the NameNode in the HDFS, in the format of <code>hdfs://ServerIP:Port</code> .
Extension Parameters	The extension parameter <code>hadoopConfig</code> for HDFS Reader and Writer. You can configure the advanced parameters of Hadoop, such as those related to HA.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.26. Configure an FTP connection

An FTP connection allows you to read data from and write data to FTP by using FTP Reader and Writer. You can configure sync nodes for FTP by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).

- ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
 3. In the **Add Connection** dialog box, click **FTP** in the Semi-Structured Storage Systems section.
 4. In the **Add FTP Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
Protocol	The protocol used by the FTP server. Only FTP and SFTP are supported.
Host	The address of the FTP server.
Port	The port of the FTP server. The default port is 21 for FTP and 22 for SFTP.
Username	The username that you can use to connect to the FTP server.
Password	The password that you can use to connect to the FTP server.
Enable reverse VPC access	Specifies whether to enable reverse VPC access. Select the Enable check box if you cannot directly access the data store on an ECS instance but can access it by using a VPC.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.27. Configure a MongoDB connection

MongoDB is a document-oriented database that is second only to Oracle and MySQL. A MongoDB connection allows you to read data from and write data to MongoDB by using MongoDB Reader and Writer. You can configure sync nodes for MongoDB by using the code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.

- iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **MongoDB** in the NoSQL section.
4. In the **Add MongoDB Connection** dialog box, set the parameters as required. You can set the **Connect To** parameter to **ApsaraDB for RDS** or **Connection Mode** for a MongoDB connection.
 - o **ApsaraDB for RDS:** Generally, the classic network is used to access the target ApsaraDB for MongoDB instance in this mode. You can access the ApsaraDB for MongoDB instance in the same region over the classic network. However, the access to the ApsaraDB for MongoDB instance from a different region over the classic network is not guaranteed to be successful.

Parameter	Description
Connect To	The connection type. In this example, set the value to ApsaraDB for RDS . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note If you have not assigned the default role to Data Integration, log on to the Resource Access Management (RAM) console with your Apsara Stack tenant account and perform authorization. Then, refresh this configuration page. </div>
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> ? Note This parameter is available only when the workspace is in standard mode. </div>
Region	The region where the ApsaraDB for MongoDB instance resides.
Instance ID	The ID of the ApsaraDB for MongoDB instance. You can view the ID in the ApsaraDB for MongoDB console.
Database Name	The name of the database that you created in the ApsaraDB for MongoDB console. You can also specify the database username and password in the console.
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

- o **Connection Mode:** Generally, the Internet is used to access the target database in this mode, which may cost you fees.

Parameter	Description
-----------	-------------

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Address	The endpoint in the host:port format. To add an endpoint, click Add Address and specify the endpoint to add. To add more endpoints, repeat the preceding action.  Note You must add either public endpoints or internal endpoints. Do not mix public endpoints with internal endpoints.
Database Name	The name of the database.
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.28. Configure a Memcache connection

A Memcache connection allows you to write data to ApsaraDB for Memcache by using Memcache Writer. You can configure sync nodes for ApsaraDB for Memcache by using the code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Memcache(OCS)** in the NoSQL section.
4. In the **Add Memcache(OCS) Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
Proxy Host	The IP address of the host or Memcache proxy. You can view the IP address on the basic information page of the ApsaraDB for Memcache console.
Port	The port for connecting to the ApsaraDB for Memcache instance. Default value: 11211.
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.29. Configure a Redis connection

A Redis connection allows you to read data from and write data to Redis by using Redis Reader and Writer. You can configure sync nodes for Redis by using the code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **Redis** in the NoSQL section.
4. In the **Add Redis Connection** dialog box, set the parameters as required. You can set the **Connect To** parameter to **ApsaraDB for RDS** or **Connection Mode** for a Redis connection.
 - o The following table describes the parameters that appear after you set the **Connect To** parameter to **ApsaraDB for RDS**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to ApsaraDB for RDS .

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Region	The region where the ApsaraDB for Redis instance resides.
Redis Instance ID	The ID of the ApsaraDB for Redis instance. You can view the ID in the ApsaraDB for Redis console.
Redis Password	The password that you can use to connect to the ApsaraDB for Redis instance. Leave it blank if no password is required.

- o The following table describes the parameters that appear after you set the **Connect To** parameter to **Connection Mode**.

Parameter	Description
Connect To	The type of the connection. In this example, set the value to Connection Mode .
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Server Address	The server address in the <code>host:port</code> format.
Add Server Address	Click Add Server Address to add a server address in the format of <code>host:port</code> .
Redis Password	The password that you can use to connect to Redis.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.30. Configure a Tablestore connection

Tablestore is a NoSQL database service built on Apsara distributed operating system. It allows you to store and access large amounts of structured data in real time.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **OTS** in the NoSQL section.
4. In the **Add OTS Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> Note This parameter is available only when the workspace is in standard mode.</div>
Endpoint	The endpoint of the Tablestore service.
Table Store Instance ID	The name of the Tablestore instance.
AccessKey ID	The AccessKey ID for connecting to the Tablestore instance. You can view the AccessKey ID on the User Info page.
AccessKey Secret	The AccessKey secret for connecting to the Tablestore instance.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.31. Configure an Elasticsearch connection

An Elasticsearch connection allows you to read data from and write data to Elasticsearch by using Elasticsearch Reader and Writer. You can configure sync nodes for Elasticsearch by using the code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).

- ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. On the **Data Source** page, click **New data source** in the upper-right corner.
 3. In the **Add Connection** dialog box, click **ElasticSearch** in the NoSQL section.
 4. In the **Add ElasticSearch Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production .  Note This parameter is available only when the workspace is in standard mode.
Endpoint	The endpoint of Elasticsearch, in the format of <code>http://esxxxx.elasticsearch.aliyuncs.com:9200</code> .
Username	The username that you can use to connect to the database.
Password	The password that you can use to connect to the database.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.4.32. Configure a LogHub connection

A LogHub connection allows you to read data from and write data to LogHub by using LogHub Reader and Writer. You can configure sync nodes for LogHub by using the codeless UI or code editor.

Procedure

1. Go to the **Data Source** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the page that appears, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
2. Click **New data source** in the upper-right corner.
3. In the **Add Connection** dialog box, click **LogHub** in the Message Queue section.
4. In the **Add LogHub Connection** dialog box, set the parameters as required.

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_). It must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
Applicable Environment	The environment in which the connection is used. Valid values: Development and Production . <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Note This parameter is available only when the workspace is in standard mode.</p> </div>
LogHub Endpoint	The LogHub endpoint, in the format of <code>http://cn-shanghai.log.aliyun.com</code> .
Project	The name of the LogHub project.
AccessKey ID	The AccessKey ID for connecting to the LogHub project. You can view the AccessKey ID on the User Info page.
AccessKey Secret	The AccessKey secret for connecting to the LogHub project.

5. Click **Test connectivity**.
6. After the data source passes the connectivity test, click **Complete**.

2.4.5. Configure data synchronization tasks

2.4.5.1. Configure a sync node by using the codeless UI

This topic describes how to configure a sync node by using the codeless user interface (UI).

To configure a sync node, follow these steps:

1. Add connections.
2. Create a sync node.
3. Select a source connection.
4. Select a destination connection.
5. Map the fields in the source and destination tables.
6. Configure the channel, such as the maximum transmission rate and dirty data check rules.
7. Configure the node properties.

 **Note** The following sections describe the overall procedure. You can click the links in each step to read relevant instructions and then return to the current page to proceed with subsequent steps.

Add connections

Data synchronization is supported between various homogenous and heterogeneous connections. Before you configure a sync node, add required connections in Data Integration. Added connections are listed as options when you configure a sync node. For more information about connection types supported by Data Integration, see [Supported data sources](#).

You can add connections of supported types to Data Integration. For more information about how to add a connection, see [Data sources](#).

 **Note**

- Data Integration does not support connectivity testing for some connection types. For more information, see [Test data store connectivity](#).
- Some connections are hosted on the premises. They do not have public IP addresses or network connections cannot be directly established. Such connections will fail the connectivity test. Data Integration allows you to add a custom resource group to resolve these issues. However, if you create sync nodes for such connections, you can only use the code editor. This is because you cannot obtain information such as table schema on the codeless UI if the network connection is unavailable.

Create a sync node

 **Note** This topic describes how to create and configure a sync node by using the codeless UI. Do not switch to the code editor.

1. Log on to the DataWorks console.
2. On the **Data Analytics** tab, move the pointer over the **Create** icon and select **Workflow**.
3. In the **Create Workflow** dialog box that appears, set **Workflow Name** and **Description**. Then, click **Create**.
4. In the left-side navigation pane, click the created workflow. Then, right-click **Data Integration** and choose **Create Data Integration Node > Sync**. In the **Create Node** dialog box that appears, set **Node Name**.
5. Click **Commit**.

Select a source connection

After the sync node is created, configure the source connection and source table.

 **Note**

- For more information about how to configure the source connection, see [Configure the reader](#).
- Incremental data synchronization is required when you configure the source connection for some sync nodes. In this case, you can use the parameter configuration feature of DataWorks to obtain the date and time required by incremental data synchronization.

Select a destination connection

After the source connection is configured, configure the destination connection and destination table.

 **Note**

- For more information about how to configure the destination connection, see [Configure the writer](#).
- You can select the writing method for most nodes. For example, the writing method can be overwriting or appending. Supported writing methods vary with the connection type.

Map the fields in the source and destination tables

After the source and destination connections are configured, specify the mapping between the fields in the source and destination tables. You can click **Map Fields with the Same Name**, **Map Fields in the Same Line**, **Delete All Mappings**, and **Auto Layout**.

Button or icon	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout. The fields are automatically sorted based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box that appears, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. You can use scheduling parameters, such as \${bizdate}. You can enter functions supported by relational databases, such as now() and count(1). Fields that cannot be parsed are indicated by Unidentified.

 **Note** Make sure that the data type of a source field is the same as or compatible with that of the mapped destination field.

Configure channel control policies

When the preceding steps are completed, configure the channel control policies of the corresponding sync node.

Parameter	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure the node properties

This section describes how to use scheduling parameters for data filtering.

On the sync node configuration tab, click the **Properties** tab in the right-side navigation pane.

You can declare the scheduling parameters by using `${Variable name}`. After a variable is declared, enter the initial value of the variable in the Arguments field. In this example, the initial value of the variable is identified by `$$`. The content can be a time expression or a constant.

For example, if you write `#{today}` in the code and enter `today=${yyyymmdd}` in the Arguments field, the value of the time variable is the current date. For more information about how to add and subtract the date, see [Parameter configuration](#).

On the Properties tab, you can configure the properties of the sync node, such as the recurrence, scheduled time, and dependencies. Sync nodes have no ancestor nodes because their corresponding jobs are run before extract, transform, and load (ETL) jobs. We recommend that you specify the root node as their parent node.

Use custom scheduling parameters

To use custom scheduling parameters for the sync node, declare the following parameters in the code:

- `bizdate`: the timestamp of data to be used by the node. The value is one day before the running date of the node.
- `cyctime`: the time when the node is run, in the format of `yyyymmddhhmiss`.
- DataWorks provides the `bizdate` and `cyctime` parameters as default system parameters.

After the sync node is configured, save and commit the node.

2.4.5.2. Configure a sync node by using the code editor

This topic describes how to configure a sync node by using the code editor.

To configure a sync node, follow these steps:

1. Add connections.
2. Create a sync node.
3. Apply a template.
4. Configure the reader.
5. Configure the writer.
6. Map the fields in the source and destination tables.
7. Configure the channel, such as the maximum transmission rate and dirty data check rules.
8. Configure the node properties.

Add connections

Data synchronization is supported between various homogenous and heterogeneous connections. Before you configure a sync node, add required connections in Data Integration. Added connections are listed as options when you configure a sync node. For more information about connection types supported by Data Integration, see [Supported data sources](#).

You can add connections of supported types to Data Integration. For more information about how to add a connection, see [Data sources](#).

 **Note** Some connections are hosted on the premises. They do not have public IP addresses or network connections cannot be directly established. Such connections will fail the connectivity test. Data Integration allows you to add a custom resource group to resolve these issues. However, if you create sync nodes for such connections, you can only use the code editor. This is because you cannot obtain information such as table schema on the codeless user interface (UI) if the network connection is unavailable.

Create a sync node

 **Note** This topic describes how to create a sync node by using the codeless UI and configure the sync node by using the code editor.

1. Log on to the DataWorks console.
2. On the **Data Analytics** tab, move the pointer over the **Create** icon and select **Workflow**.
3. In the **Create Workflow** dialog box that appears, set **Workflow Name** and **Description**. Then, click **Create**.
4. In the left-side navigation pane, click the created workflow. Then, right-click **Data Integration** and choose **Create Data Integration Node > Sync**. In the **Create Node** dialog box that appears, set **Node Name**.
5. Click **Commit**.

Apply a template

1. After the sync node is created, the node configuration tab appears. Click the **Switch to Code Editor** icon in the toolbar.
2. In the **Confirm** dialog box that appears, click **OK** to switch to the code editor.

 **Note** The code editor supports more features than the codeless UI. For example, you can configure sync nodes in the code editor even when the connectivity test fails.

3. Click the **Apply Template** icon in the toolbar.
4. In the **Apply Template** dialog box that appears, set **Source Connection Type**, **Connection**, **Target Connection Type**, and **Connection**.
5. Click **OK**.

Configure the reader

After the template is applied, the basic settings of the reader are configured. You can configure the source connection and source table as needed.

```

{"type": "job",
 "version": "2.0",
 "steps": [
  {
    "stepType": "mysql", // The reader type.
    "parameter": {
      "datasource": "MySQL", // The connection name.
      "column": [ // The columns to be synchronized.
        "id",
        "value",
        "table"
      ],
      "socketTimeout": 3600000, // The timeout period for reading data from and writing data to a socket, in milliseconds.
      "connection": [
        {
          "datasource": "MySQL", // The connection name.
          "table": [
            "` case ` " // The name of the table to be synchronized.
          ]
        }
      ],
      "where": "", // The WHERE clause.
      "splitPk": "", // The shard key.
      "encoding": "UTF-8" // The encoding format.
    },
    "name": "Reader",
    "category": "reader" // Indicates that these settings are related to the reader.
  },
 ]
}

```

The parameters are described as follows:

- type: the type of the sync node. You must set the value to job.
- version: the version number of the sync node. You can set the value to 1.0 or 2.0.

Note

- For more information about how to configure the source connection in the code editor, see [Configure the reader](#).
- Incremental data synchronization is required when you configure the source connection for some sync nodes. In this case, you can use the parameter configuration feature of DataWorks to obtain the date and time required by incremental data synchronization.

Configure the writer

After the reader is configured, you can configure the destination connection and destination table as needed.

```
{
  "stepType": "odps", // The writer type.
  "parameter": {
    "partition": "", // The partitions that the reader reads.
    "truncate": true, // Specifies whether to clear up previous data and import new data when a write operation is performed again after failure. Set the value to true to guarantee the idempotence of write operations.
    "compress": false, // Specifies whether to enable compression.
    "datasource": "odps_first", // The connection name.
    "column": [ // The columns to be synchronized.
      "*"
    ],
    "emptyAsNull": false,
    "table": ""
  },
  "name": "Writer",
  "category": "writer" // Indicates that these settings are related to the writer.
}
```

 **Note**

- For more information about how to configure the destination connection in the code editor, see [Configure the writer](#).
- You can select the writing method for most nodes. For example, the writing method can be overwriting or appending. Supported writing methods vary with the connection type.

Map the fields in the source and destination tables

The code editor only supports mapping of fields in the same row. Note that the data types of the fields must match.

 **Note** Make sure that the data type of a source field is the same as or compatible with that of the mapped destination field.

Configure channel control policies

When the preceding steps are completed, configure the channel control policies of the corresponding sync node. The setting parameter specifies the node efficiency, including the settings on the DUM number, thread concurrency, bandwidth throttling, dirty data policy, and resource group.

```
"setting": {
  "errorLimit": {
    "record": "1024" // The maximum number of dirty data records allowed.
  },
  "speed": {
    "throttle": false, // Specifies whether to enable bandwidth throttling.
    "concurrent": 1, // The maximum number of concurrent threads.
  }
},
```

Setting	Description
---------	-------------

Setting	Description
Expected concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node in the code editor.
Bandwidth throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty data records allowed	The maximum number of dirty data records allowed.
Resource group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure the node properties

This section describes how to use scheduling parameters for data filtering.

On the sync node configuration tab, click the **Properties** tab in the right-side navigation pane.

On the Properties tab, you can configure the properties of the sync node, such as the recurrence, scheduled time, and dependencies. Sync nodes have no ancestor nodes because their corresponding jobs are run before extract, transform, and load (ETL) jobs. We recommend that you specify the root node as their parent node.

After the sync node is configured, save and commit the node.

2.4.5.3. Configure the reader

2.4.5.3.1. Configure DRDS Reader

Distributed Relational Database Service (DRDS) Reader allows you to read data from DRDS. DRDS Reader connects to a remote DRDS database and runs a SELECT statement to select and read data from the database.

Currently, DRDS Reader only supports MySQL engines. DRDS is a distributed MySQL database service that complies with MySQL protocols in most cases.

Specifically, DRDS Reader connects to a remote DRDS database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and then sends the statement to the database. The DRDS database runs the statement and returns the result. Then, DRDS Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

DRDS Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SELECT statement to the DRDS database. DRDS does not support all MySQL specifications, such as JOIN statements.

DRDS Reader supports most DRDS data types. Make sure that your data types are supported.

The following table lists the data types supported by DRDS Reader.

Category	DRDS data type
Integer	INT, TINYINT, SMALLINT, MEDIUMINT, and BIGINT
Floating point	FLOAT, DOUBLE, and DECIMAL
String	VARCHAR, CHAR, TINYTEXT, TEXT, MEDIUMTEXT, and LONGTEXT
Date and time	DATE, DATETIME, TIMESTAMP, TIME, and YEAR
Boolean	BIT and BOOLEAN
Binary	TINYBLOB, MEDIUMBLOB, BLOB, LONGBLOB, and VARBINARY

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized.	Yes	None
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select and export specific columns. Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in compliance with the SQL syntax supported by MySQL. for example. <code>["id", "`table`", "1", "bazhen.csy", "null", "to_char(a + 1)", "2.3", "true"]</code>. <ul style="list-style-type: none"> id: a column name. table: the name of a column that contains reserved keywords. 1: an integer constant. bazhen.csy: a string constant. null: a null pointer. to_char(a + 1): a function expression. 2.3: a floating-point constant. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. 	Yes	None

Parameter	Description	Required	Default value
where	<p>The WHERE clause. DRDS Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to <code>STRTODATE('\${bdp.system.bizdate}' . '%Y%m%d') <= today AND today < DATEADD(STRTODATE('\${bdp.system.bizdate}', '%Y%m%d'), interval 1 day) .</code></p> <ul style="list-style-type: none"> You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. 	No	None

Configure DRDS Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Filter	The filter condition for the data to be synchronized. Currently, filtering based on the limit keyword is not supported. The SQL syntax is determined by the selected connection.
Shard Key	<p>The shard key. You can use a column in the source table as the shard key. We recommend that you use the primary key or an indexed column. Only integer fields are supported.</p> <p>If data sharding is performed based on the configured shard key, data can be read concurrently to improve data synchronization efficiency.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note The Shard Key parameter is displayed only when you configure the source connection for a sync node.</p> </div>

2. Configure field mapping, that is, the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Parameter	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.

Parameter	Description
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout. The fields are automatically sorted based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box that appears, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	Click Add to add a field. The rules for adding fields are described as follows: <ul style="list-style-type: none"> You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. You can use scheduling parameters, such as \${bizdate}. You can enter functions supported by relational databases, such as now() and count(1). Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless user interface (UI).
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure DRDS Reader by using the code editor

In the following code, a node is configured to read data from a DRDS database.

```

{
  "type":"job",
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"drds", // The reader type.
      "parameter":{
        "datasource":""," // The connection name.
        "column":[ // The columns to be synchronized.
          "id",
          "name"
        ],
        "where":""," // The WHERE clause.
        "table":""," // The name of the table to be synchronized.
        "splitPk":""," // The shard key.
      },
      "name":"Reader",
      "category":"reader"
    },
    // The following template is used to configure Stream Writer. For more information about how to configure other writers, see the corresponding topic.
    {
      "stepType":"stream", // The writer type.
      "parameter":{
        "name":"Writer",
        "category":"writer"
      }
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false, // Specifies whether to enable bandwidth throttling.
      "concurrent":1, // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}:"Writer"
}
}
}
}

```

Additional instructions

- Consistency

As a distributed database service, DRDS cannot provide a consistent view of multiple tables in multiple databases. Different from MySQL where data is synchronized in a single table of a single database, DRDS Reader cannot extract the snapshot of database and table shards at the same time slice. That is, DRDS Reader extracts different snapshots from different shards. As a result, this cannot guarantee strong consistency for data queries.

- Character encoding

DRDS supports flexible encoding configurations. You can specify the encoding format for an instance, a field, a table, and a database. The configurations for the field, table, database, and instance are prioritized in descending order. We recommend that you use UTF-8 for a database.

DRDS Reader uses JDBC, which can automatically convert the encoding of characters. Therefore, you do not need to specify the encoding format.

If you specify the encoding format for a DRDS database but data is written to the DRDS database in a different encoding format, DRDS Reader cannot recognize this inconsistency and may export garbled characters.

- Incremental data synchronization

DRDS Reader connects to a database through JDBC and uses a SELECT statement with a WHERE clause to read incremental data in the following ways:

- For data in batches, incremental add, update, and delete operations (including logically delete operations) are distinguished by timestamps. Specify the WHERE clause based on the timestamp. The timestamp must be later than the latest timestamp in the last synchronization.
- For streaming data, specify the WHERE clause based on the data record ID. The data record ID must be larger than the maximum ID involved in the last synchronization.

If incremental data cannot be distinguished, DRDS Reader cannot perform incremental synchronization but can perform full synchronization only.

- Syntax validation

DRDS Reader allows you to specify custom SELECT statements by using the querySql parameter but does not verify the syntax of the custom SELECT statements.

2.4.5.3.2. Configure HBase Reader

HBase Reader allows you to read data from HBase. HBase Reader connects to a remote HBase database through a Java client of HBase. Then, HBase Reader scans and reads data based on the specified rowkey range, assembles the data to abstract datasets in custom data types supported by Data Integration, and then passes the datasets to a writer.

Data types

The following table lists the data types supported by HBase Reader.

Category	Data Integration data type	HBase data type
Integer	LONG	Short, Int, and Long
Floating point	DOUBLE	Float and Double
String	STRING	Binary_String and String
Date and time	DATE	Date
Byte	BYTES	Bytes

Category	Data Integration data type	HBase data type
Boolean	BOOLEAN	Boolean

Parameters

Parameter	Description	Required	Default value
haveKerberos	<p>Specifies whether Kerberos authentication is required. A value of true indicates that Kerberos authentication is required.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #d9e1f2;"> <p>Note</p> <ul style="list-style-type: none"> • If the value is true, the following five Kerberos-related parameters must be specified: <ul style="list-style-type: none"> ◦ kerberosKeytabFilePath ◦ kerberosPrincipal ◦ hbaseMasterKerberosPrincipal ◦ hbaseRegionserverKerberosPrincipal ◦ hbaseRpcProtection • If the value is false, Kerberos authentication is not required and you do not need to specify the preceding parameters. </div>	No	<i>false</i>
hbaseConfig	The properties of the HBase cluster, in JSON format. The hbase.zookeeper.quorum parameter is required. It specifies the ZooKeeper ensemble servers. You can also configure other properties, such as those related to the cache and batch for scan operations.	Yes	None
mode	The mode in which data is read from the HBase connection. Valid values: normal and multiVersionFixedColumn.	Yes	None
table	The name of the HBase table from which data is read. The name is case-sensitive.	Yes	None
encoding	The encoding format, by using which binary data stored in byte[] format is converted into strings. Currently, UTF-8 and GBK are supported.	No	UTF-8

Parameter	Description	Required	Default value
column	<p>The HBase columns from which data is read.</p> <ul style="list-style-type: none"> In normal mode: <p>The name parameter specifies the name of the column in the HBase table. The format must be columnFamily:columnName except for the rowkey. The type parameter specifies the source data type. The format parameter specifies the date format. The value parameter specifies the column value if the column is a constant column. Example:</p> <pre> "column": [{ "name": "rowkey", "type": "string" }, { "value": "test", "type": "string" }] </pre> <p>For the column parameter, you must specify the type parameter and specify one of the name and value parameters.</p> <ul style="list-style-type: none"> In multiVersionFixedColumn mode: <p>The name parameter specifies the name of the column in the HBase table. The format must be columnFamily:columnName except for the rowkey. The type parameter specifies the source data type. The format parameter specifies the date format. You cannot create constant columns in multiVersionFixedColumn mode. Example:</p> <pre> "column": [{ "name": "rowkey", "type": "string" }, { "name": "info:age", "type": "string" }] </pre> 	Yes	None
maxVersion	<p>The number of versions read by HBase Reader when multiple versions are available. Valid values: -1 and integers greater than 1. A value of -1 indicates that all versions are read.</p>	Required in multiVersionFixedColumn mode	None

Parameter	Description	Required	Default value
range	<p>The rowkey range that HBase Reader reads.</p> <ul style="list-style-type: none"> startRowkey: the start rowkey. endRowkey: the end rowkey. isBinaryRowkey: the method used to convert the specified start and end rowkeys into the byte[] format. Default value: false. If the value is true, Bytes.toBytesBinary(rowkey) is used. If the value is false, Bytes.toBytes(rowkey) is used. Example: <pre>"range":{ "startRowkey": "aaa", "endRowkey": "ccc", "isBinaryRowkey":false }</pre>	No	None
scanCacheSize	The number of rows read by an HBase client with each remote procedure call (RPC) connection.	No	256
scanBatchSize	The number of columns read by an HBase client with each RPC connection.	No	100

Configure HBase Reader by using the codeless UI

Currently, the codeless user interface (UI) is not supported for HBase Reader.

Configure HBase Reader by using the code editor

In the following code, a node is configured to read data from an HBase connection in normal mode.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "hbase", // The reader type.
      "parameter": {
        "mode": "normal",
        "scanCacheSize": 256, // The number of rows read by an HBase client with each RPC connection.
        "scanBatchSize": 256, // The number of columns read by an HBase client with each RPC connection.
        "hbaseVersion": "094x",
        "datasource": "demo_hbase", // The connection name.
        "column": [
          {
            "name": "info:idx",
            "type": "long"
          },
          {
            "name": "info:age",
            "type": "string"
          },
          {
            "name": "info:birthday",
            "format": "yyyy-MM-dd",
            "type": "date"
          }
        ]
      }
    }
  ]
}
```

```
    },
    ],
    "range": {
      "startRowKey": "", // The start rowkey.
      "endRowKey": "", // The end rowkey.
      "isBinaryRowKey": false // The method used to convert the specified start and end rowkeys into the byte[]
      format. Default value: false. If the value is true, Bytes.toBytesBinary(rowkey) is used. If the value is false, Bytes.toBy
      tes(rowkey) is used.
    },
    "maxVersion": , // The number of versions read by HBase Reader when multiple versions are available. Valid v
    alues: -1 and integers greater than 1. A value of -1 indicates that all versions are read.
    "encoding": "UTF-8",
    "table": "test" // The name of the HBase table from which data is read. The name is case-sensitive.
  },
  "name": "Reader",
  "category": "reader"
},
"stepType": "odps", // The writer type.
"parameter": {},
"name": "Writer",
"category": "writer"
}
],
"setting": {
},
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}
```

In the following code, a node is configured to read data from an HBase connection in multiVersionFixedColumn mode.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "hbase", // The reader type.
      "parameter": {
        "table": "users", // The name of the HBase table from which data is read. The name is case-sensitive.
        "encoding": "utf-8", // The encoding format, by using which binary data stored in byte[] format is converte
        d into strings. Currently, UTF-8 and GBK are supported.
        "mode": "multiVersionFixedColumn",
        "maxVersion": "-1", // The number of versions read by HBase Reader when multiple versions are available.
        Valid values: -1 and integers greater than 1. A value of -1 indicates that all versions are read.
        "column": [ // The HBase columns from which data is read. The name parameter specifies the name of the c
        olumn in the HBase table. The format must be columnFamily:columnName except for the rowkey. The type parame
        ter specifies the source data type. The format parameter specifies the date format. You cannot create constant colu
        mns in multiVersionFixedColumn mode.
        {
          "name": "rowkey",
          "type": "string"
        }
      ]
    }
  ]
}
```

```

        type: "string"
      },
      {
        "name": "info: age",
        "type": "string"
      },
      {
        "name": "info: birthday",
        "type": "date",
        "format": "yyyy-MM-dd"
      }
    ],
    "range": { // The rowkey range that HBase Reader reads.
      "startRowkey": "",
      "endRowkey": ""
    }
  },
  "name": "Reader",
  "category": "reader"
},
{
  "stepType": "odps", // The writer type.
  "parameter": {},
  "name": "Writer",
  "category": "writer"
}
],
"setting": {
},
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}
}
}

```

2.4.5.3.3. Configure HDFS Reader

HDFS Reader allows you to read data stored in a Hadoop Distributed File System (HDFS). HDFS Reader connects to an HDFS, reads data from files in the HDFS, converts the data into a format that is readable by Data Integration, and then sends the converted data to a writer.

Examples:

Text File is the default storage format for creating Hive tables, without data compression. Essentially, a Text File file is stored in HDFS as text. For Data Integration, the implementation of HDFS Reader is similar to that of OSS Reader.

Optimized Row Columnar File (ORCFile) is an optimized RCF file format. It provides an efficient method for storing Hive data. HDFS Reader uses the OrcSerde class provided by Hive to read and parse ORCFile data.

 **Note**

- Considering that a complex network connection is required between the default resource group and HDFS, we recommend that you use a custom resource group to run sync nodes. Make sure that your custom resource group can access the NameNode and DataNode of HDFS through a network.
- By default, HDFS uses a network whitelist to guarantee data security. In this case, we recommend that you use a custom resource group to run HDFS sync nodes.
- If you configure an HDFS sync node in the code editor, the HDFS connection does not need to pass the connectivity test. In this case, you can temporarily ignore connectivity test errors.
- To synchronize data in Data Integration, you must log on as an administrator. Make sure that you have the permissions to read data from and write data to relevant HDFS files.

Features

Currently, HDFS Reader supports the following features:

- Supports the TextFile, ORCFile, RCFile, SequenceFile, CSV, and Parquet file formats. What is stored in each file must be a logical two-dimensional table.
- Reads data of various types as strings. Supports constants and column pruning.
- Supports recursive reading. Supports regular expressions that contain asterisks (*) and question marks (?).
- Compresses ORCFile files in SNAPPY or ZLIB format.
- Compresses SequenceFile files in LZO format.
- Reads multiple files concurrently.
- Compresses CSV files in GZIP, BZIP2, ZIP, LZO, LZO_DEFLATE, or SNAPPY format.
- Supports Hive 1.1.1 and Hadoop 2.7.1 (compatible with Apache JDK 1.6). HDFS Reader can work properly with Hadoop 2.5.0, Hadoop 2.6.0, and Hive 1.2.0 during testing.

 **Note** Currently, HDFS Reader cannot use concurrent threads to read a single file.

Data types

RCFile

RCFile metadata is stored in databases managed by Hive, and in different formats depending on the data type. However, HDFS Reader cannot query metadata from such databases. If you want to synchronize a file of the RCFile format, you must specify the data type for each column. If the data type is BIGINT, DOUBLE, or FLOAT, specify the data type as BIGINT, DOUBLE, or FLOAT. If the data type is VARCHAR or CHAR, specify the data type as STRING.

RCFile data types are automatically converted into the data types supported by Data Integration. The following table lists the supported data types.

Category	HDFS data type
Integer	TINYINT, SMALLINT, INT, and BIGINT
Floating point	FLOAT, DOUBLE, and DECIMAL
String	STRING, CHAR, and VARCHAR
Date and time	DATE and TIMESTAMP
Boolean	BOOLEAN

Category	HDFS data type
Binary	BINARY

Parquet files

Parquet file data types are automatically converted into the data types supported by Data Integration. The following table lists the supported data types.

Category	HDFS data type
Integer	INT32, INT64, and INT96
Floating point	FLOAT and DOUBLE
String	FIXED_LEN_BYTE_ARRAY
Date and time	DATE and TIMESTAMP
Boolean	BOOLEAN
Binary	BINARY

Text File, ORCFile, and SequenceFile

Text File metadata and ORCFile metadata are stored in databases, such as MySQL databases, managed by Hive. However, HDFS Reader cannot query metadata from such databases. If you want to convert data types during data synchronization, you must specify the data types.

Text File, ORCFile, and SequenceFile data types are automatically converted into the data types supported by Data Integration. The following table lists the supported data types.

Category	HDFS data type
Integer	TINYINT, SMALLINT, INT, and BIGINT
Floating point	FLOAT and DOUBLE
String	STRING, CHAR, VARCHAR, STRUCT, MAP, ARRAY, UNION, and BINARY
Date and time	DATE and TIMESTAMP
Boolean	BOOLEAN

The data types are described as follows:

- **LONG:** integer strings in HDFS files, such as 123456789.
- **DOUBLE:** double value strings in HDFS files, such as 3.1415.
- **BOOLEAN:** Boolean strings in HDFS files, such as true and false. The strings are case-insensitive.
- **DATE:** date and time strings in HDFS files, such as 2014-12-31 00:00:00.

 **Note** The TIMESTAMP data type of Hive is accurate to nanoseconds. If you convert TIMESTAMP-type Hive data, such as 2015-08-21 22:40:47.397898389, in Text File and ORCFile files into the DATE type in Data Integration, the converted data is accurate to seconds. If you need nanosecond-scale accuracy, convert TIMESTAMP-type data into the STRING type in Data Integration.

Parameters

Parameter	Description	Required	Default value
path	<p>The path of the file to read. To read multiple files, use a regular expression such as /hadoop/data_201704*.</p> <ul style="list-style-type: none"> If you specify a single HDFS file, HDFS Reader uses only one thread to read the file. If you specify multiple HDFS files, HDFS Reader uses multiple threads. The number of threads is limited by the transmission rate, in Mbit/s. <p>Note The actual number of threads is determined by both the number of HDFS files to be read and the specified transmission rate.</p> <ul style="list-style-type: none"> When a path contains a wildcard, HDFS Reader attempts to read all files that match the path. If the path is ended with a slash (/), HDFS Reader reads all files in the specified directory. For example, if you specify the path as /bazhen/, HDFS Reader reads all files in the bazhen directory. Currently, HDFS Reader only supports asterisks (*) and question marks (?) as file name wildcards. The syntax is similar to that of file name wildcards used on the Linux command line. <p>Note</p> <ul style="list-style-type: none"> Data Integration considers all the files on a sync node as a single table. Make sure that all the files on each sync node can adapt to the same schema and Data Integration has the permission to read all these files. Note: When creating Hive tables, you can specify partitions. For example, if you specify partition(day="20150820",hour="09"), a directory named /20150820 and a subdirectory named /09 are created in the corresponding table directory of the HDFS. <p>Therefore, if you need HDFS Reader to read the data of a partition, specify the file path of the partition. For example, if you need HDFS Reader to read all the data in the partition with the date of 20150820 in the table named mytable01, specify the path as follows:</p> <pre>"path": "/user/hive/warehouse/mytable01/20150820/*"</pre>	Yes	None
defaultFS	<p>The address of the NameNode of the HDFS. If a sync node is run on the default resource group, advanced parameter settings of Hadoop, such as those related to high availability, are not supported.</p>	Yes	None

Parameter	Description	Required	Default value
fileType	<p>The file format. Valid values: text, orc, rc, seq, csv, and parquet. HDFS Reader automatically recognizes the file format and uses corresponding read policies. Before data synchronization, HDFS Reader checks whether all the source files match the specified format. If any source file does not match the format, the sync node fails.</p> <p>The valid values of the fileType parameter are described as follows:</p> <ul style="list-style-type: none"> • text: the TextFile format. • orc: the ORCFile format. • rc: the RCFile format. • seq: the SequenceFile format. • csv: the common HDFS file format, that is, the logical two-dimensional table. • parquet: the common Parquet file format. <div style="background-color: #e1f5fe; padding: 10px; border: 1px solid #cfe2f3;"> <p> Note</p> <p>TextFile and ORCFile are different formats. HDFS Reader parses files in the two formats in different ways. After being converted from a composite data type of Hive into the STRING type of Data Integration, the data in a file of the TextFile format can be different from that in the same file of the ORCFile format. Composite data types include MAP, ARRAY, STRUCT, and UNION. The following example uses the conversion from the MAP type to the STRING type as an example:</p> <ul style="list-style-type: none"> • HDFS Reader converts MAP-type ORCFile data into a string: {job=80, team=60, person=70}. • HDFS Reader converts MAP-type TextFile data into a string: job:80, team:60, person:70. <p>The conversion results show that the data remains unchanged but the formats differ slightly. Therefore, if the data to be synchronized matches a composite data type of Hive, we recommend that you use a uniform file format.</p> </div> <p>Recommendations:</p> <ul style="list-style-type: none"> • To use a uniform file format, we recommend that you export TextFile tables as ORCFile tables on the Hive client. • If the file format is Parquet, the parquetSchema parameter is required, which specifies the schema of the Parquet table. <p>For the column parameter, you must specify the type parameter and specify one of the index and value parameters.</p>	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to read. The type parameter specifies the source data type. The index parameter specifies the ID of the column in the source table, starting from 0. The value parameter specifies the column value if the column is a constant column. By default, HDFS Reader reads all data as strings. Specify this parameter as <code>"column":["*"]</code>.</p> <p>You can also specify the column parameter in the following way:</p> <pre>{ "type": "long", "index": 0 // The first INT-type column of the source file. }, { "type": "string", "value": "alibaba" // The value of the current column, that is, a constant "alibaba". }</pre>	Yes	None
fieldDelimiter	<p>The column delimiter. To read TextFile data, you must specify the column delimiter. The default delimiter is comma (,). To read ORCFile data, you do not need to specify the column delimiter. The default delimiter is <code>\u0001</code>.</p> <ul style="list-style-type: none"> If you need each row to be converted into a column in the destination table, use a string that does not exist in every row, such as <code>\u0001</code>. Do not use <code>\n</code> as the delimiter. 	No	,
encoding	The encoding format of the file to read.	No	UTF-8
nullFormat	<p>The string that represents null. No standard strings can represent null in text files. Therefore, Data Integration provides the nullFormat parameter to define which string represents a null pointer.</p> <p>For example, if you specify nullFormat:"null", Data Integration considers null as a null pointer.</p>	No	None
compress	<p>The compression format. Available compression formats for CSV files are GZIP, BZIP2, ZIP, LZO, LZO_DEFLATE, and SNAPPY.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> Do not mix up LZO with LZO_DEFLATE. Snappy does not have a uniform stream format. Data Integration currently only supports the most popular two compression formats: hadoop-snappy (Snappy stream format in Hadoop) and framing-snappy (Snappy stream format recommended by Google). rc indicates the RCFile format. This parameter is not required for files of the ORCFile format. </div>	No	None

Parameter	Description	Required	Default value
parquetSchema	<p>The schema of the source file. This parameter is required only when the fileType parameter is set to parquet. Format:</p> <pre data-bbox="405 383 1114 533">message messageType { required, dataType, columnName; ; }</pre> <p>The format is described as follows:</p> <ul data-bbox="405 613 1114 808" style="list-style-type: none"> • messageType: the name of the MessageType object. • required: specifies whether the field is required or optional. We recommend that you set the parameter to optional for all fields. • dataType: the data type of the field. Supported data types: BOOLEAN, INT32, INT64, INT96, FLOAT, DOUBLE, BINARY, and FIXED_LEN_BYTE_ARRAY. Select BINARY if the data type is STRING. <div data-bbox="405 831 1114 936" style="background-color: #e0f2f7; padding: 5px;"> <p> Note Each line, including the last one, must end with a semicolon (;).</p> </div> <p>An example is provided as follows:</p> <pre data-bbox="405 1003 1114 1368">message m { optional int64 id; optional int64 date_id; optional binary datetimestring; optional int32 dspId; optional int32 advertiserId; optional int32 status; optional int64 bidding_req_num; optional int64 imp; optional int64 click_num; }</pre>	No	None

Parameter	Description	Required	Default value
csvReaderConfig	<p>The configurations for reading CSV files. The parameter value must match the MAP type. A specific CSV reader is used to read data from CSV files, which supports many configurations.</p> <p>The following example provides common configurations:</p> <pre>"csvReaderConfig":{ "safetySwitch": false, "skipEmptyRecords": false, "useTextQualifier": false }</pre> <p>You can use the following parameters and their default values:</p> <pre>boolean caseSensitive = true; char textQualifier = 34; boolean trimWhitespace = true; boolean useTextQualifier = true; // Specifies whether to use escape characters for CSV files. char delimiter = 44; // The delimiter. char recordDelimiter = 0; char comment = 35; boolean useComments = false; int escapeMode = 1; boolean safetySwitch = true; // Specifies whether to limit the length of each column to 100,000 characters. boolean skipEmptyRecords = true; // Specifies whether to skip empty rows. boolean captureRawRecord = true;</pre>	No	None
hadoopConfig	<p>The advanced parameter settings of Hadoop, such as those related to high availability.</p> <pre>"hadoopConfig":{ "dfs.nameservices": "testDfs", "dfs.ha.namenodes.testDfs": "namenode1,namenode2", "dfs.namenode.rpc-address.youkuDfs.namenode1": "", "dfs.namenode.rpc-address.youkuDfs.namenode2": "", "dfs.client.failover.proxy.provider.testDfs": "org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider" }</pre>	No	None

Configure HDFS Reader by using the codeless UI

Currently, the codeless user interface (UI) is not supported for HDFS Reader.

Configure HDFS Reader by using the code editor

In the following code, a node is configured to read data from an HDFS. For more information about the parameters, see the preceding parameter description.

```


```

```

{
  "type": "job",
  "version": "2.0",
  "steps": [
    {
      "stepType": "hdfs", // The reader type.
      "parameter": {
        "path": "", // The path of the file to read.
        "datasource": "", // The connection name.
        "column": [
          {
            "index": 0, // The ID of the column in the source table.
            "type": "string" // The data type.
          },
          {
            "index": 1,
            "type": "long"
          },
          {
            "index": 2,
            "type": "double"
          },
          {
            "index": 3,
            "type": "boolean"
          },
          {
            "format": "yyyy-MM-dd HH:mm:ss", // The format of the time.
            "index": 4,
            "type": "date"
          }
        ],
        "fieldDelimiter": ",", // The column delimiter.
        "encoding": "UTF-8", // The encoding format.
        "fileType": "" // The file format.
      },
      "name": "Reader",
      "category": "reader"
    },
    // The following template is used to configure the writer. For more information, see the corresponding topic.
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "concurrent": 3, // The maximum number of concurrent threads.
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    }
  },
  "order": {

```

```
"hops": [
  {
    "from": "Reader",
    "to": "Writer"
  }
]
```

2.4.5.3.4. Configure MaxCompute Reader

This topic describes the data types and parameters supported by MaxCompute Reader and how to configure it by using the codeless UI and code editor.

MaxCompute Reader can read data from a MaxCompute project by using the MaxCompute Tunnel service based on the source project, table, partition, and table fields that you have configured.

MaxCompute Reader cannot read views. It can read only partitioned tables and non-partitioned tables. To allow MaxCompute Reader to read partitioned tables, you must specify the partition information. For example, set `pt` to 1 and `ds` to `hangzhou` for the `t0` table. The partition information is not required for non-partitioned tables. Additionally, you can select some or all of the table fields, change the order in which the fields are arranged, or add constant fields and partition key columns. Note that partition key columns are not table fields.

Data types

The following table lists the data types supported by MaxCompute Reader.

Category	Data Integration data type	MaxCompute data type
Integer	LONG	BIGINT, INT, TINYINT, and SMALLINT
Boolean	BOOLEAN	BOOLEAN
Date and time	DATE	DATETIME and TIMESTAMP
Floating point	DOUBLE	FLOAT, DOUBLE, and DECIMAL
Binary	BYTES	BINARY
Complex	STRING	ARRAY, MAP, and STRUCT

Parameters

Parameter	Description	Required	Default value
<code>datasource</code>	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
<code>table</code>	The name of the source table. The name is not case-sensitive.	Yes	None

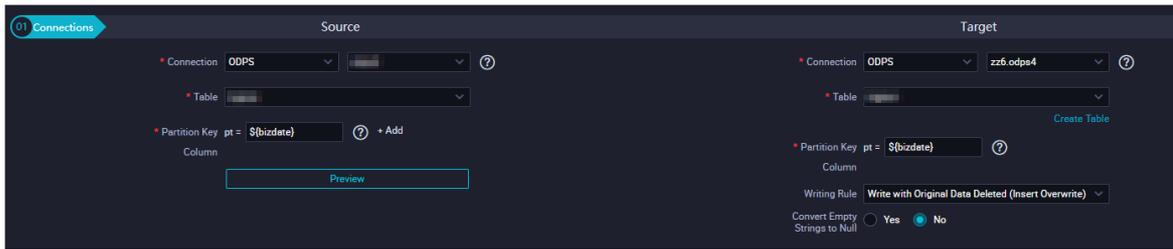
Parameter	Description	Required	Default value
partition	<p>The partitions that MaxCompute Reader reads. Linux shell wildcards are supported. An asterisk (*) represents zero or more characters, and a question mark (?) represents that the previous character can be included or not. Assume that a partitioned table named test has four partitions: pt=1 and ds=hangzhou, pt=1 and ds=shanghai, pt=2 and ds=hangzhou, and pt=2 and ds=beijing.</p> <ul style="list-style-type: none"> To read data from the partition with pt=1 and ds=shanghai, enter <code>"partition":"pt=1/ds=shanghai"</code>. To read data from all the partitions with pt=1, enter <code>"partition":"pt=1/ds=*"</code>. To read data from all the partitions in the test table, enter <code>"partition":"pt=*/ds=*"</code>. 	Required only for writing data to a partitioned table	None
column	<p>The columns in the source table that MaxCompute Reader reads. Assume that the fields of a table named test are id, name, and age.</p> <ul style="list-style-type: none"> To read the fields in turn, enter <code>"column":["id","name","age"]</code> or <code>"column":["*"]</code>. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 5px 0;"> <p>Note We recommend that you do not set <code>"column":["*"]</code>. This is because data synchronization may fail if the source table changes in the column order, data type, or number of columns.</p> </div> <ul style="list-style-type: none"> To read the name and id fields in turn, enter <code>"column":["name","id"]</code>. You can add a constant field to extracted data for the purpose of proper mapping between source table columns and destination table columns. Each constant must be enclosed in single quotation marks (' '). For example, if you set <code>"column":["age","name","1988-08-08 08:08:08","id"]</code>, the data extracted contains an age column, a name column, a constant "1988-08-08 08:08:08", and an id column in turn. <p>The single quotation marks (' ') are used to identify constant columns. The constant column values exclude the single quotation marks (' ').</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 5px 0;"> <p>Note</p> <ul style="list-style-type: none"> MaxCompute Reader does not use SELECT statements to read data. Therefore, you cannot specify function fields. The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. </div>	Yes	None

Configure MaxCompute Reader by using the codeless UI

On the **DataStudio** page, create a sync node under a workflow and configure the node.

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.



Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Partition Key Column	The partition information. You can click Add on the right to add partition key columns.

Note To synchronize all columns in the source table, enter "column":["*"]. The partition parameter supports wildcards and includes one or more partitions.

- "partition": "pt=20140501/ds=*" specifies that all ds partitions with pt=20140501 are to be synchronized.
- "partition": "pt=top?" specifies that the partitions with pt=top and pt=to are to be synchronized.

You can specify the partition key columns to be synchronized, such as a partition key column named pt. Assume that the partition key column of a MaxCompute table is pt=\${bdp.system.bizdate}. You can configure the column to be synchronized to pt. Ignore it if the column is marked as unidentified. To synchronize all partitions, enter pt=*. To synchronize specified partitions, specify the corresponding dates.

2. Configure field mapping. It is equivalent to setting the column parameter provided in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

GUI element	Description
Change Fields	Click the Change Fields icon. In the Change Fields dialog box, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. ◦ You can use scheduling parameters, such as \${bizdate}. ◦ You can enter functions supported by relational databases, such as now() and count(1). ◦ Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure MaxCompute Reader by using the code editor

In the following code, a node is configured to read data from MaxCompute. For more information about the parameters, see the preceding parameter description.

```
{
  "type":"job", // The type of the sync node.
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"odps", // The reader type.
      "parameter":{
        "partition":[], // The partitions that MaxCompute Reader reads.
        "isCompress":false, // Specifies whether to enable compression.
        "datasource":""," // The connection name.
        "column":[// The columns to be synchronized.
          "id"
        ],
        "emptyAsNull":true,
        "table":""," // The table name.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"stream", // The writer type.
      "parameter":{ // The parameters that you specify for the writer.
      },
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1 // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader", // The source connection of the node.
        "to":"Writer" // The destination connection of the node.
      }
    ]
  }
}
```

2.4.5.3.5. Configure MongoDB Reader

This topic describes the data types and parameters supported by MongoDB Reader and how to configure it by using the code editor.

MongoDB Reader connects to a remote MongoDB database by using the Java client named MongoClient and reads data from the database. The latest version of MongoDB has improved the locking feature from database locks to document locks. By using the powerful functionalities of indexes in MongoDB, MongoDB Reader can efficiently read data from MongoDB databases.

Note

- If you use ApsaraDB for MongoDB, the MongoDB database has a root account by default. For security concerns, Data Integration supports access to a MongoDB database only by using a MongoDB database account. When you add a MongoDB connection, do not use the root account for access.
- JavaScript syntax is not supported for queries.

MongoDB Reader shards data in the MongoDB database based on specified rules, reads data from the database with multiple threads, and then converts the data to a format readable by Data Integration.

Data types

MongoDB Reader supports most MongoDB data types. Make sure that your data types are supported.

The following table lists the data types supported by MongoDB Reader.

Category	MongoDB data type
Long	INT, LONG, DOCUMENT.INT, and DOCUMENT.LONG
Double	DOUBLE and DOCUMENT.DOUBLE
String	STRING, ARRAY, DOCUMENT.STRING, DOCUMENT.ARRAY, and COMBINE
Date	DATE and DOCUMENT.DATE
Boolean	BOOLEAN and DOCUMENT.BOOLEAN
Bytes	BYTES and DOCUMENT.BYTES

Note

- The DOCUMENT data type is used to store embedded documents. It is also called the OBJECT data type.
- The following content describes how to use the COMBINE data type:

When MongoDB Reader reads data from a MongoDB database, it combines and converts multiple fields in MongoDB documents to a JSON string.

For example, doc1, doc2, and doc3 are three MongoDB documents with different fields, which are represented by keys instead of key-value pairs. The keys a and b represent common fields in all the three documents. The key x_n represents an unfixed field.

```
doc1: a b x_1 x_2
doc2: a b x_2 x_3 x_4
doc3: a b x_5
```

To import the preceding three MongoDB documents to MaxCompute, you must specify the fields to retain, set a name for each combined string, and set the data type of each combined string to COMBINE in the configuration file. Make sure that the name of each combined string is unique among all existing fields in the documents.

```
"column": [
  {
    "name": "a",
    "type": "string",
  },
  {
    "name": "b",
    "type": "string",
  },
  {
    "name": "doc",
    "type": "combine",
  }
]
```

The following table lists the output in MaxCompute.

odps_column1	odps_column2
a	b
a	b
a	b

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
collectionName	The name of the replica set in MongoDB.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns in MongoDB.</p> <ul style="list-style-type: none"> name: the name of the column. type: the data type of the column. splitter: the delimiter. Specify this parameter only when you need to convert the string to an array. MongoDB supports arrays, but Data Integration does not. The array elements read by MongoDB are joined to a string by using this delimiter. 	Yes	None
query	<p>The filter condition for obtaining data from MongoDB. Only the time type is supported. For example, you can use the statement</p> <pre>"query":{"!operationTime": {\$gte:ISODate('\${last_day}T00:00:00.424+0800')}}"</pre> <p>to obtain data where the time specified by operationTime is not earlier than 00:00 on the day specified by \${last_day}. In the preceding JSON string, \${last_day} is a scheduling parameter of DataWorks. The format is \${yyyy-mm-dd}. You can use comparison operators (such as \$gt, \$lt, \$gte, and \$lte), logical operators (such as \$and and \$or), and functions (such as max, min, sum, avg, and ISODate) supported by MongoDB as needed.</p>	No	None

Configure MongoDB Reader by using the codeless UI

The codeless UI is not supported for MongoDB Reader.

Configure MongoDB Reader by using the code editor

In the following code, a node is configured to read data from a MongoDB database. For more information about the parameters, see the preceding parameter description.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    "reader":{
      "plugin": "mongodb",// The reader type.
      "parameter":{
        "datasource": "datasourceName",// The connection name.
        "collectionName": "tag_data",// The name of the MongoDB collection.
        "query":"","
        "column": [
          {
            "name": "unique_id", // The field name.
            "type": "string" // The data type.
          },
          {
            "name": "sid",
            "type": "string"
          },
          {
            "name": "user_id",
            "type": "string"
          },
          {
            "name": "auction_id",
```

```
    "type": "string"
  },
  {
    "name": "content_type",
    "type": "string"
  },
  {
    "name": "pool_type",
    "type": "string"
  },
  {
    "name": "frontcat_id",
    "type": "array",
    "splitter": ""
  },
  {
    "name": "categoryid",
    "type": "array",
    "splitter": ""
  },
  {
    "name": "gmt_create",
    "type": "string"
  },
  {
    "name": "taglist",
    "type": "array",
    "splitter": ""
  },
  {
    "name": "property",
    "type": "string"
  },
  {
    "name": "scorea",
    "type": "int"
  },
  {
    "name": "scoreb",
    "type": "int"
  },
  {
    "name": "scorec",
    "type": "int"
  },
  {
    "name": "a.b",
    "type": "document.int"
  },
  {
    "name": "a.b.c",
    "type": "document.array",
    "splitter": ""
  }
]
}
```

```

{
  "stepType":"stream",
  "parameter":{},
  "name":"Writer",
  "category":"writer"
}
],
"setting":{
  "errorLimit":{
    "record":"0"// The maximum number of dirty data records allowed.
  },
  "speed":{
    "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent":1,// The maximum number of concurrent threads.
  }
},
"order":{
  "hops":[
    {
      "from":"Reader",
      "to":"Writer"
    }
  ]
}
}
}

```

 **Note** You cannot retrieve data elements from arrays.

2.4.5.3.6. Configure Db2 Reader

This topic describes the data types and parameters supported by Db2 Reader and how to configure it by using the code editor.

Db2 Reader allows you to read data from Db2. Db2 Reader connects to a remote Db2 database and runs a SELECT statement to select and read data from the database.

Specifically, Db2 Reader connects to a remote Db2 database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and then sends the statement to the database. The Db2 database runs the statement and returns the result. Then, Db2 Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

- Db2 Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SELECT statement to the Db2 database.
- If you specify the querySql parameter, Db2 Reader directly sends the value of this parameter to the Db2 database.

Db2 Reader supports most Db2 data types. Make sure that your data types are supported.

The following table lists the data types supported by Db2 Reader.

Category	Db2 data type
Integer	SMALLINT

Category	Db2 data type
Floating point	DECIMAL, REAL, and DOUBLE
String	CHAR, CHARACTER, VARCHAR, GRAPHIC, VARGRAPHIC, LONG VARCHAR, CLOB, LONG VARGRAPHIC, and DBCLOB
Date and time	DATE, TIME, and TIMESTAMP
Boolean	N/A
Binary	BLOB

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
jdbcUrl	The JDBC URL for connecting to the Db2 database. In accordance with official Db2 specifications, the URL must be in the <code>jdbc:db2://ip:port/database</code> format. You can also specify the information of the attachment facility.	Yes	None
username	The username for connecting to the database.	Yes	None
password	The password for connecting to the database.	Yes	None
table	The name of the table to be synchronized. You can select only one source table for each sync node.	Yes	None
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select and export specific columns. Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in compliance with the SQL syntax supported by Db2. For example, <code>["id", "1", "const name", "null", "upper('abc_lower')", "2.3", "true"]</code>. <ul style="list-style-type: none"> id: a column name. 1: an integer constant. 'const name': a string constant, which is enclosed in single quotation marks (' '). null: a null pointer. upper('abc_lower'): a function expression. 2.3: a floating-point constant. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. 	Yes	None

Parameter	Description	Required	Default value
splitPk	<p>The field used for data sharding when Db2 Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. Currently, the splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you specify this parameter to a column of an unsupported type, Db2 Reader returns an error. 	No	""
where	<p>The WHERE clause. Db2 Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to <code>gmt_create>\$bizdate</code>. You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized.</p>	No	None
querySql	<p>The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter.</p> <p>For example, if you want to join multiple tables for data synchronization, set this parameter to <code>select a,b from table_a join table_b on table_a.id = table_b.id</code>. If you specify the querySql parameter, Db2 Reader ignores the table, column, and where parameters that you have configured.</p>	No	None
fetchSize	<p>The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects reading efficiency.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note A value greater than 2048 may lead to out of memory (OOM) during the data synchronization process.</p> </div>	No	1024

Configure Db2 Reader by using the codeless UI

Currently, the codeless user interface (UI) is not supported for Db2 Reader.

Configure Db2 Reader by using the code editor

In the following code, a node is configured to read data from a Db2 database.

```
{
  "type":"job",
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"db2", // The reader type.
      "parameter":{"
        "password":""," // The password for connecting to the database.
        "jdbcUrl":""," // The JDBC URL for connecting to the Db2 database.
        "column":[
          "id"
        ],
        "where":""," // The WHERE clause.
        "splitPk":""," // The field used for data sharding. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter.
        "table":""," // The name of the table to be synchronized.
        "username":""," // The username for connecting to the database.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{"
    "errorLimit":{"
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{"
      "throttle":false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1, // The maximum number of concurrent threads.
    }
  },
  "order":{"
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

Additional instructions

- Data synchronization between primary and secondary databases

A secondary Db2 database can be deployed for disaster recovery. The secondary database continuously synchronizes data from the primary database based on binlogs. Especially when network conditions are unfavorable, data latency between the primary and secondary databases is unavoidable, which can lead to data inconsistency.

- **Concurrency control**

Db2 is a relational database management system (RDBMS), which supports strong consistency for data queries. A database snapshot is created before a sync node starts. Db2 Reader reads data from the database snapshot. Therefore, if new data is written to the database during data synchronization, Db2 Reader cannot obtain the new data.

Data consistency cannot be guaranteed when you enable Db2 Reader to run concurrent threads on a single sync node.

Db2 Reader shards the table based on the `splitPk` parameter and runs multiple concurrent threads to synchronize data. These concurrent threads belong to different transactions. They read data at different time points. This means that the concurrent threads observe different snapshots.

Theoretically, the data inconsistency issue is unavoidable if a single sync node includes multiple threads. However, two workarounds are available:

- Do not enable concurrent threads on a single sync node. Essentially, do not specify the `splitPk` parameter. In this way, data consistency is guaranteed although data is synchronized at a low efficiency.
- Disable writers to make sure that the data is unchanged during data synchronization. For example, lock the table and disable data synchronization between primary and secondary databases. In this way, data is synchronized efficiently but your ongoing services may be interrupted.

- **Character encoding**

Db2 Reader uses JDBC, which can automatically convert the encoding of characters. Therefore, you do not need to specify the encoding format.

- **Incremental data synchronization**

Db2 Reader connects to a database through JDBC and uses a SELECT statement with a WHERE clause to read incremental data in the following ways:

- For data in batches, incremental add, update, and delete operations (including logically delete operations) are distinguished by timestamps. Specify the WHERE clause based on the timestamp. The timestamp must be later than the latest timestamp in the last synchronization.
- For streaming data, specify the WHERE clause based on the data record ID. The data record ID must be larger than the maximum ID involved in the last synchronization.

If incremental data cannot be distinguished, Db2 Reader cannot perform incremental synchronization but can perform full synchronization only.

- **Syntax validation**

Db2 Reader allows you to specify custom SELECT statements by using the `querySql` parameter but does not verify the syntax of the custom SELECT statements.

2.4.5.3.7. Configure MySQL Reader

This topic describes the data types and parameters supported by MySQL Reader and how to configure it by using the codeless user interface (UI) and code editor.

MySQL Reader connects to a remote MySQL database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and then sends the statement to the database. The MySQL database runs the statement and returns the result. Then, MySQL Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

In short, MySQL Reader connects to a remote MySQL database and runs a SELECT statement to select and read data from the database.

MySQL Reader can read tables and views. For table fields, you can specify all or some of the columns in sequence, adjust the column order, specify constant fields, and configure MySQL functions, such as `now()`.

Data types

The following table lists the data types supported by MySQL Reader.

Category	MySQL data type
Integer	INT, TINYINT, SMALLINT, MEDIUMINT, and BIGINT
Floating point	FLOAT, DOUBLE, and DECIMAL
String	VARCHAR, CHAR, TINYTEXT, TEXT, MEDIUMTEXT, and LONGTEXT
Date and time	DATE, DATETIME, TIMESTAMP, TIME, and YEAR
Boolean	BIT and BOOLEAN
Binary	TINYBLOB, MEDIUMBLOB, BLOB, LONGBLOB, and VARBINARY

Note

- Data types that are not listed in the table are not supported.
- MySQL Reader considers tinyint(1) as the INTEGER type.
- Currently, MySQL Reader does not support MySQL 8.0 or later.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized. You can select only one source table for each sync node.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select and export specific columns. Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in compliance with the SQL syntax supported by MySQL. For example, <code>["id"."table"."1","mingya.wmy","null","to_char(a+1)","2.3","true"]</code>. <ul style="list-style-type: none"> id: a column name. table: the name of a column that contains reserved keywords. 1: an integer constant. 'mingya.wmy': a string constant, which is enclosed in single quotation marks (' '). null: <ul style="list-style-type: none"> " " indicates an empty value. null indicates a null value. 'null' indicates the string null. to_char(a + 1): a function expression. 2.3: a floating-point constant. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when MySQL Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. Currently, the splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you specify this parameter to a column of an unsupported type, MySQL Reader ignores the splitPk parameter and synchronizes data through a single thread. If you do not specify the splitPk parameter or leave it empty, Data Integration synchronizes data through a single thread. 	No	None

Parameter	Description	Required	Default value
where	<p>The WHERE clause. For example, set this parameter to <code>gmt_create>\$bizdate</code> .</p> <ul style="list-style-type: none"> You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. Do not set the where parameter to limit 10, which does not conform to the constraints of MySQL on the SQL WHERE clause. 	No	None
querySql (only available in the code editor)	<p>The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter. For example, if you want to join multiple tables for data synchronization, set this parameter to <code>select a.b from table_a join table_b on table_a.id = table_b.id</code> . The priority of the querySql parameter is higher than those of the table, column, where, and splitPk parameters. If you specify the querySql parameter, MySQL Reader ignores the table, column, where, and splitPk parameters that you have configured. The datasource parameter parses information, including the username and password, from this parameter.</p>	No	None
singleOrMulti (applicable only to database and table sharding)	<p>Specifies whether to shard the database or table. After you switch from the codeless UI to the code editor, the following configuration is automatically generated: <code>"singleOrMulti": "multi"</code> . However, if you use the code editor since the beginning, the configuration is not automatically generated and you must manually specify this parameter. If you do not specify this parameter or leave it empty, MySQL Reader can only read data from the first shard.</p>	Yes	<i>multi</i>

Configure MySQL Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Filter	The filter condition for the data to be synchronized. Currently, filtering based on the limit keyword is not supported. The SQL syntax is determined by the selected connection.

Parameter	Description
Shard Key	<p>The shard key. You can use a column in the source table as the shard key. We recommend that you use the primary key or an indexed column. Only integer fields are supported.</p> <p>If data sharding is performed based on the configured shard key, data can be read concurrently to improve data synchronization efficiency.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note The Shard Key parameter is displayed only when you configure the source connection for a sync node.</p> </div>

2. Configure field mapping, that is, the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Parameter	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout. The fields are automatically sorted based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box that appears, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. ◦ You can use scheduling parameters, such as \${bizdate}. ◦ You can enter functions supported by relational databases, such as now() and count(1). ◦ Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.

Parameter	Description
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure MySQL Reader by using the code editor

In the following code, a node is configured to read data from a database or table that is not sharded. For more information about the parameters, see the preceding parameter description.

```

{
  "type":"job",
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"mysql", // The reader type.
      "parameter":{
        "column":[ // The columns to be synchronized.
          "id"
        ],
        "connection":[
          { "querySql":["select a,b from join1 c join join2 d on c.id = d.id;"], // Specify the querySql parameter in the connection parameter as a string.
            "datasource":""," // The connection name.
            "table":[
              "xxx" // The name of the table to be synchronized.
            ]
          }
        ],
        "where":""," // The WHERE clause.
        "splitPk":""," // The shard key.
        "encoding":"UTF-8" // The encoding format.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1, // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}

```

In the following code, a node is configured to read data from a database or table that is sharded. For more information about the parameters, see the preceding parameter description.

Note In the case of database and table sharding, MySQL Reader can read multiple MySQL tables with the same schema.

```
{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "reader": {
      "plugin": "mysql",
      "parameter": {
        "connection": [
          {
            "table": [
              "tbl1",
              "tbl2",
              "tbl3"
            ],
            "datasource": "datasourceName1"
          },
          {
            "table": [
              "tbl4",
              "tbl5",
              "tbl6"
            ],
            "datasource": "datasourceName2"
          }
        ],
        "singleOrMulti": "multi",
        "splitPk": "db_id",
        "column": [
          "id", "name", "age"
        ],
        "where": "1 < id and id < 100"
      }
    },
    "writer": {
    }
  }
}
```

2.4.5.3.8. Configure Oracle Reader

This topic describes the data types and parameters supported by Oracle Reader and how to configure it by using the codeless user interface (UI) and code editor.

Oracle Reader allows you to read data from Oracle. Oracle Reader connects to a remote Oracle database and runs a SELECT statement to select and read data from the database.

Specifically, Oracle Reader connects to a remote Oracle database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and then sends the statement to the database. The Oracle database runs the statement and returns the result. Then, Oracle Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

- Oracle Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SELECT statement to the Oracle database.
- If you specify the querySql parameter, Oracle Reader directly sends the value of this parameter to the Oracle database.

Data types

Oracle Reader supports most Oracle data types. Make sure that your data types are supported.

The following table lists the data types supported by Oracle Reader.

Category	Oracle data type
Integer	NUMBER, ROWID, INTEGER, INT, and SMALLINT
Floating point	NUMERIC, DECIMAL, FLOAT, DOUBLE PRECISION, and REAL
String	LONG, CHAR, NCHAR, VARCHAR, VARCHAR2, NVARCHAR2, CLOB, NCLOB, CHARACTER, CHARACTER VARYING, CHAR VARYING, NATIONAL CHARACTER, NATIONAL CHAR, NATIONAL CHARACTER VARYING, NATIONAL CHAR VARYING, and NCHAR VARYING
Date and time	TIMESTAMP and DATE
Boolean	BIT and BOOLEAN
Binary	BLOB, BFILE, RAW, and LONG RAW

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select and export specific columns. Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in JSON format. <pre>["id", "1", "mingya.wmy", "null", "to_char(a + 1)", "2.3", "true"]</pre> <ul style="list-style-type: none"> id: a column name. 1: an integer constant. 'mingya.wmy': a string constant, which is enclosed in single quotation marks (' '). null: a null pointer. to_char(a + 1): a function expression. 2.3: a floating-point constant. true: a Boolean value. <ul style="list-style-type: none"> The column parameter must be specified. 	Yes	None
splitPk	<p>The field used for data sharding when Oracle Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. The data types supported by the splitPk parameter include INTEGER, STRING, FLOAT, and DATE. If you do not specify the splitPk parameter or leave it empty, Oracle Reader synchronizes data through a single thread. 	No	None
where	<p>The WHERE clause. Oracle Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to row_number() or <code>id>2 and sex=1</code>.</p> <ul style="list-style-type: none"> You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. 	No	None

Parameter	Description	Required	Default value
querySql (only available in the code editor)	The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter. For example, if you want to join multiple tables for data synchronization, set this parameter to <code>select a,b from table_a join table_b on table_a.id = table_b.id</code> . If you specify the querySql parameter, Oracle Reader ignores the table, column, and where parameters that you have configured.	No	None
fetchSize	The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects reading efficiency. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Note A value greater than 2048 may lead to out of memory (OOM) during the data synchronization process.</p> </div>	No	1024

Configure Oracle Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Filter	The filter condition for the data to be synchronized. Currently, filtering based on the limit keyword is not supported. The SQL syntax is determined by the selected connection.
Shard Key	The shard key. You can use a column in the source table as the shard key. We recommend that you use the primary key or an indexed column. Only integer fields are supported. If data sharding is performed based on the configured shard key, data can be read concurrently to improve data synchronization efficiency. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Note The Shard Key parameter is displayed only when you configure the source connection for a sync node.</p> </div>

2. Configure field mapping, that is, the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Parameter	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout. The fields are automatically sorted based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box that appears, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. ◦ You can use scheduling parameters, such as \${bizdate}. ◦ You can enter functions supported by relational databases, such as now() and count(1). ◦ Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure Oracle Reader by using the code editor

In the following code, a node is configured to read data from an Oracle database.

```

{
  "type":"job",
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"oracle",
      "parameter":{
        "fetchSize":1024, // The number of data records to read at a time.
        "datasource":"", // The connection name.
        "column":[ // The columns to be synchronized.
          "id",
          "name"
        ],
        "where":"", // The WHERE clause.
        "splitPk":"", // The shard key.
        "table":"" // The name of the table to be synchronized.
      },
      "name":"Reader",
      "category":"reader"
    },
    // The following template is used to configure Stream Writer. For more information about how to configure other writers, see the corresponding topic.
    {
      "stepType":"stream",
      "parameter":{
        "name":"Writer",
        "category":"writer"
      }
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1, // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
} "to":"Writer"
}
}
}
}

```

Additional instructions

- Data synchronization between primary and secondary databases

A secondary Oracle database can be deployed for disaster recovery. The secondary database continuously synchronizes data from the primary database based on binlogs. Especially when network conditions are unfavorable, data latency between the primary and secondary databases is unavoidable, which can lead to data inconsistency.

- Concurrency control

Oracle is a relational database management system (RDBMS), which supports strong consistency for data queries. A database snapshot is created before a sync node starts. Oracle Reader reads data from the database snapshot. Therefore, if new data is written to the database during data synchronization, Oracle Reader cannot obtain the new data.

Data consistency cannot be guaranteed when you enable Oracle Reader to run concurrent threads on a single sync node.

Oracle Reader shards the table based on the `splitPk` parameter and runs multiple concurrent threads to synchronize data. These concurrent threads belong to different transactions. They read data at different time points. This means that the concurrent threads observe different snapshots.

Theoretically, the data inconsistency issue is unavoidable if a single sync node includes multiple threads. However, two workarounds are available:

- Do not enable concurrent threads on a single sync node. Essentially, do not specify the `splitPk` parameter. In this way, data consistency is guaranteed although data is synchronized at a low efficiency.
- Disable writers to make sure that the data is unchanged during data synchronization. For example, lock the table and disable data synchronization between primary and secondary databases. In this way, data is synchronized efficiently but your ongoing services may be interrupted.

- Character encoding

Oracle Reader uses JDBC, which can automatically convert the encoding of characters. Therefore, you do not need to specify the encoding format.

- Incremental data synchronization

Oracle Reader connects to a database through JDBC and uses a `SELECT` statement with a `WHERE` clause to read incremental data in the following ways:

- For data in batches, incremental add, update, and delete operations (including logically delete operations) are distinguished by timestamps. Specify the `WHERE` clause based on the timestamp. The timestamp must be later than the latest timestamp in the last synchronization.
- For streaming data, specify the `WHERE` clause based on the data record ID. The data record ID must be larger than the maximum ID involved in the last synchronization.

If incremental data cannot be distinguished, Oracle Reader cannot perform incremental synchronization but can perform full synchronization only.

- Syntax validation

Oracle Reader allows you to specify custom `SELECT` statements by using the `querySql` parameter but does not verify the syntax of the custom `SELECT` statements.

2.4.5.3.9. Configure OSS Reader

This topic describes the data types and parameters supported by OSS Reader and how to configure it by using the codeless user interface (UI) and code editor.

OSS Reader can read data stored in OSS. Specifically, OSS Reader connects to Object Storage Service (OSS) by using Alibaba Cloud OSS SDK for Java and reads data from OSS. Then, OSS Reader converts the data into a format that is readable to Data Integration and sends the converted data to a writer.

OSS stores only unstructured data. OSS Reader supports the following features:

- Reads TXT objects that store two-dimensional tables. OSS Reader can read only TXT objects.
- Reads data stored in CSV-like objects with custom delimiters.
- Reads various types of data as strings and supports constants and column pruning.
- Supports recursive data read and object name-based filtering.
- Supports the following object compression formats: GZIP, BZIP2, and ZIP.

 **Note** You cannot compress multiple objects into one package.

- Uses parallel threads to read multiple objects.

OSS Reader does not support the following features:

- Uses parallel threads to read an uncompressed object.
- Uses parallel threads to read a compressed object.

OSS Reader supports the following OSS data types: BIGINT, DOUBLE, STRING, DATETIME, and BOOLEAN.

Parameters

Parameter	Description	Required	Default value
datasource	The name of the data source. It must be the same as the name of the added data source. You can add data sources by using the code editor.	Yes	No default value
Object	<p>The name of the OSS object from which you want to read data. You can specify multiple object names. For example, a bucket has a directory that is named yunshi, and this directory contains an object that is named ll.txt. In this case, you can set this parameter to yunshi/ll.txt.</p> <ul style="list-style-type: none"> • If you specify a single OSS object name, OSS Reader uses only one thread to read the object. Parallel thread reading of a single uncompressed object will be available in the future. • If you specify multiple OSS object names, OSS Reader uses multiple threads to read these objects. The actual number of threads is determined by the number of channels. • If a name contains a wildcard, OSS Reader reads all objects that match the name. For example, you set this parameter to abc[0-9]. In this case, OSS Reader reads data from objects abc0 to abc9. We recommend that you do not use wildcards because wildcards may cause out of memory (OOM) errors. <p> Note</p> <ul style="list-style-type: none"> • Data Integration considers all the objects on a synchronization node as a single table. Make sure that all the objects on each synchronization node can adapt to the same schema. • Control the number of objects stored in a single directory. If a directory contains excessive objects, an OOM error may be returned. In this case, store the objects in different directories and then synchronize data. 	Yes	No default value

Parameter	Description	Required	Default value
column	<p>The columns from which you want to read data. The type parameter specifies the source data type. The index parameter specifies the ID of the column in the source object, starting from 0. The value parameter specifies the column value if the column is a constant column.</p> <p>By default, OSS Reader reads all data as strings. You can specify the column parameter in the following way:</p> <pre>json "column": ["*"]</pre> <p>You can also specify the column parameter in the following way:</p> <pre>json "column": { "type": "long", "index": 0// The first INT-type column of the source object. }, { "type": "string", "value": "alibaba"// The value of the current column. In this code, the value is the constant "alibaba". }</pre> <p>Note For the column parameter, you must specify the type parameter and specify either the index or value parameter.</p>	Yes	Asterisks (Not recommended)
fieldDelimiter	<p>The column delimiter that is used in the OSS object from which you want to read data.</p> <p>Note You must specify the column delimiter for OSS Reader. The default delimiter is commas (.). If you do not specify the column delimiter, the default column delimiter is also used on the codeless UI.</p>	Yes	,
compress	<p>The compression format of the object from which you want to read data. By default, this parameter is left empty. In this case, objects are not compressed. OSS Reader supports the following object compression formats: GZIP, BZIP2, and ZIP.</p>	No	<i>No default value</i>
encoding	<p>The encoding format of the object from which you want to read data.</p>	No	<i>utf-8</i>
nullFormat	<p>The string that represents a null pointer. No standard strings can represent a null pointer in TXT objects. Therefore, Data Integration provides the nullFormat parameter to define which string represents a null pointer. For example, if you specify <code>nullFormat="null"</code>, Data Integration considers null as a null pointer. You can use the following formula to escape empty strings: <code>\N=\\N</code>.</p>	No	No default value

Parameter	Description	Required	Default value
skipHeader	Specifies whether to skip the header of a CSV-like object if the object has a header. The skipHeader parameter is not supported for compressed objects.	No	<i>false</i>
csvReaderConfig	The configurations for reading CSV objects. The parameter value must match the MAP type. A specific CSV reader is used to read data from CSV objects. The CSV reader supports multiple configurations.	No	No default value

Configure OSS Reader by using the codeless UI

1. Configure data sources.

Configure **Source** and **Target** for the synchronization node.

Parameter	Description
Connection	This parameter corresponds to the datasource parameter that is described in the preceding section. Select the name of a data source that you configured.
Object Name Prefix	This parameter corresponds to the object parameter that is described in the preceding section. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note If an OSS object is named based on the date, such as aaa/20171024abc.txt, you can set this parameter to aaa/\${bdp.system.bizdate}abc.txt.</p> </div>
Field Delimiter	This parameter corresponds to the fieldDelimiter parameter that is described in the preceding section. The default delimiter is commas (,).
Encoding	This parameter corresponds to the encoding parameter that is described in the preceding section. The default encoding format is UTF-8.
Null String	This parameter corresponds to the nullFormat parameter that is described in the preceding section. Enter a string that represents a null pointer. If the source contains the string, the string is replaced with a null pointer.
Compression Format	This parameter corresponds to the compress parameter that is described in the preceding section. By default, objects are not compressed.
Include Header	This parameter corresponds to the skipHeader parameter that is described in the preceding section. The default value is No.

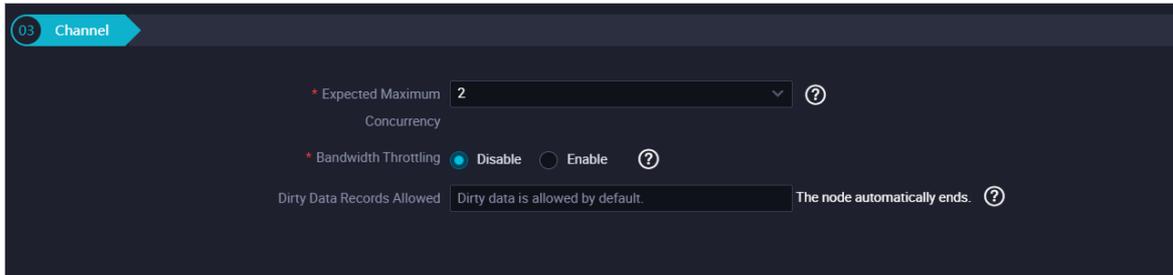
2. Configure field mappings. This operation is equivalent to setting the column parameter that is described in the preceding section.

Fields in the source on the left have a one-to-one mapping with fields in the destination on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Operation	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish mappings between fields with the same name. The data types of the fields must match.

Operation	Description
Map Fields in the Same Line	Click Map Fields in the Same Line to establish mappings between fields in the same row. The data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that are established.

3. Configure channel control policies.



Parameter	Description
Expected Maximum Concurrency	The maximum number of parallel threads that the synchronization node uses to read data from the source or write data to the destination. You can configure the parallelism for the synchronization node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workloads on the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to an appropriate value based on the configurations of the source.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.

Configure OSS Reader by using the code editor

In the following code, a synchronization node is configured to read data from OSS. For more information about the parameters, see the preceding parameter description.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "oss", // The reader type.
      "parameter": {
        "nullFormat": "", // The string that represents a null pointer.
        "compress": "", // The compression format.
        "datasource": "", // The name of the data source.
        "column": [ // The names of the columns from which you want to read data.
          {
            "index": 0, // The IDs of the columns from which you want to read data.
            "type": "string" // The data type.
          },
          {
            "index": 1,
            "type": "long"
          }
        ]
      }
    }
  ]
}
```

```

    {
      "index":2,
      "type":"double"
    },
    {
      "index":3,
      "type":"boolean"
    },
    {
      "format":"yyyy-MM-dd HH:mm:ss", // The time format.
      "index":4,
      "type":"date"
    }
  ],
  "skipHeader":""," // Specifies whether to skip the header of a CSV-like object if the object has a header.
  "encoding":""," // The encoding format.
  "fieldDelimiter":",", // The column delimiter.
  "fileFormat":""," // The format in which OSS Reader stores the object from which you want to read data.
  "object":[] // The name of the object from which you want to read data.
},
"name":"Reader",
"category":"reader"
},
{
  "stepType":"stream",
  "parameter":{},
  "name":"Writer",
  "category":"writer"
}
],
"setting":{
  "errorLimit":{
    "record":""," // The maximum number of dirty data records allowed.
  },
  "speed":{
    "throttle":false, // Specifies whether to enable bandwidth throttling. The value false indicates that bandwidth throttling is disabled, and the value true indicates that bandwidth throttling is enabled. The mbps parameter takes effect only when the throttle parameter is set to true.
    "concurrent":1, // The number of parallel threads.
  }
},
"order":{
  "hops":[
    {
      "from":"Reader",
      "to":"Writer"
    }
  ]
}
}
}

```

2.4.5.3.10. Configure FTP Reader

This topic describes the data types and parameters supported by File Transfer Protocol (FTP) Reader and how to configure it by using the codeless user interface (UI) and code editor.

FTP Reader allows you to read data from a remote FTP server. FTP Reader connects to an FTP server, reads data from the server, converts the data into a format that is readable by Data Integration, and then sends the converted data to a writer.

FTP Reader can read only FTP files that store logical two-dimensional tables, for example, text information in CSV format.

FTP servers store unstructured data only. Currently, FTP Reader supports the following features:

- Reads TXT files that store logical two-dimensional tables. FTP Reader can read only TXT files.
- Reads data stored in formats similar to CSV with custom delimiters.
- Reads data of various types as strings. Supports constants and column pruning.
- Supports recursive reading and file name-based filtering.
- Supports the following file compression formats: GZIP, BZIP2, ZIP, LZO, and LZO_DEFLATE.
- Reads multiple files concurrently.

Currently, FTP Reader does not support the following features:

- Uses concurrent threads to read an uncompressed file.
- Uses concurrent threads to read a compressed file.

The data types of remote FTP files are defined by FTP Reader.

Data Integration data type	FTP file data type
LONG	LONG
DOUBLE	DOUBLE
STRING	STRING
BOOLEAN	BOOLEAN
DATE	DATE

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None

Parameter	Description	Required	Default value
path	<p>The path of the FTP file to read. You can specify multiple FTP file paths.</p> <ul style="list-style-type: none"> If you specify a single FTP file, FTP Reader uses only one thread to read the file. Concurrent multi-thread reading of a single uncompressed file is coming soon. If you specify multiple FTP files, FTP Reader uses multiple threads to read these files. The actual number of threads is determined by the number of channels. When a path contains a wildcard, FTP Reader attempts to read all files that match the path. If the path is ended with a slash (/), FTP Reader reads all files in the specified directory. For example, if you specify the path as /bazhen/, FTP Reader reads all files in the bazhen directory. Currently, FTP Reader only supports asterisks (*) as file name wildcards. <div style="background-color: #e0f2f1; padding: 10px; margin-top: 10px;"> <p>Note</p> <ul style="list-style-type: none"> We recommend that you do not use asterisks (*) because this may cause out of memory (OOM) on a Java virtual machine (JVM). Data Integration considers all the files on a sync node as a single table. Make sure that all the files on each sync node can adapt to the same schema and Data Integration has the permission to read all these files. Make sure that the data format is similar to CSV. An error occurs if no readable files exist in the specified path. </div>	Yes	None
column	<p>The columns to read. The type parameter specifies the source data type. The index parameter specifies the ID of the column in the source table, starting from 0. The value parameter specifies the column value if the column is a constant column.</p> <p>By default, FTP Reader reads all data as strings. Specify this parameter as <code>"column":["*"]</code>. You can also specify the column parameter in the following way:</p> <pre style="background-color: #f5f5f5; padding: 10px; margin: 10px 0;"> { "type": "long", "index": 0 // The first INT-type column of the source file. }, { "type": "string", "value": "alibaba" // The value of the current column, that is, a constant "alibaba". } </pre> <p>For the column parameter, you must specify the type parameter and specify one of the index and value parameters.</p>	Yes	By default, FTP Reader reads all data as strings.

Parameter	Description	Required	Default value
fieldDelimiter	<p>The column delimiter.</p> <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e0f0ff;"> <p> Note You must specify the column delimiter for FTP Reader. The default delimiter is comma (,). The default setting for the column delimiter on the codeless UI is comma (,), too.</p> </div>	Yes	,
skipHeader	Specifies whether to skip the header (if exists) of a CSV-like file. The skipHeader parameter is not supported for compressed files.	No	false
encoding	The encoding format of the file to read.	No	UTF-8
nullFormat	<p>The string that represents null. No standard strings can represent null in text files. Therefore, Data Integration provides the nullFormat parameter to define which string represents a null pointer.</p> <p>For example, if you specify <code>nullFormat:"null"</code>, Data Integration considers null as a null pointer.</p>	No	None
markDoneFileName	The name of the file used to indicate that the sync node can start. Data Integration checks whether the file exists before data synchronization. If the file does not exist, Data Integration checks again later. Data Integration starts the sync node only after the file is detected.	No	None
maxRetryTime	The maximum number of checks for the file used to indicate that the sync node can start. By default, 60 checks are allowed. Data Integration checks for the file every 1 minute. The whole process lasts at most 60 minutes.	No	60
csvReaderConfig	The configurations for reading CSV files. The parameter value must match the MAP type. A specific CSV reader is used to read data from CSV files, which supports many configurations.	No	None
fileFormat	<p>The format of the file saved by FTP Reader. By default, FTP Reader converts the data into a two-dimensional table and stores the table in a CSV file. If you specify binary as the file format, Data Integration converts data into the binary format for replication and transmission.</p> <p>Generally, you need to specify this parameter only when you want to replicate the complete directory structure between storage systems such as FTP and Object Storage Service (OSS).</p>	No	None

Configure FTP Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.

Parameter	Description
File Path	The path parameter in the preceding parameter description.
File Type	The format of the file saved by FTP Reader. The default format is CSV.
Field Delimiter	The fieldDelimiter parameter in the preceding parameter description. The default delimiter is comma (,).
Encoding	The encoding parameter in the preceding parameter description. The default encoding format is <i>UTF-8</i> .
Null String	The nullFormat parameter in the preceding parameter description, which defines a string that represents the null value.
Compression Format	The compression format. By default, files are not compressed.
Include Header	The skipHeader parameter in the preceding parameter description. The default value is <i>No</i> .

2. Configure field mapping, that is, the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Parameter	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.

3. Configure channel control policies.

Parameter	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure FTP Reader by using the code editor

In the following code, a node is configured to read data from an FTP server.

```
{
  "type":"job",
  "version":"2.0", // The version number.
  "steps":[
    {
      "stepType":"ftp", // The reader type.
      "parameter":{
        "path":[], // The file path.
        "nullFormat":"", // The string that represents null.
        "compress":"", // The compression format.
        "datasource":"", // The connection name.
        "column":[ // The columns to be synchronized.
          {
            "index":0, // The ID of the column in the source table.
            "type":"" // The data type.
          }
        ],
        "skipHeader":"", // Specifies whether to skip the file header.
        "fieldDelimiter":",", // The column delimiter.
        "encoding":"UTF-8", // The encoding format.
        "fileFormat":"csv" // The format of the file saved by FTP Reader.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      // The following template is used to configure the writer. For more information, see the corresponding topic.
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1, // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

2.4.5.3.11. Configure Tablestore Reader

This topic describes the data types and parameters supported by Tablestore Reader and how to configure it by using the code editor.

Tablestore Reader can read incremental data from Tablestore based on the specified range. Tablestore Reader can read incremental data in the following ways:

- Reads data from the entire table.
- Reads data based on the specified range.
- Reads data from the specified shard.

Tablestore is a NoSQL database service that is built on the Apsara distributed operating system. The service allows you to store and access large volumes of structured data in real time. Tablestore organizes data into instances and tables. It uses data sharding and load balancing technologies to seamlessly expand the data scale.

Tablestore Reader connects to the Tablestore server by using Tablestore SDK for Java and reads data from the server. Then, Tablestore Reader converts the data into a format that is readable to Data Integration based on the official data synchronization protocols, and sends the converted data to a writer.

Tablestore Reader splits a synchronization node into multiple concurrent tasks based on the table range to synchronize data in a Tablestore table. Each Tablestore Reader thread runs a task.

Tablestore Reader supports all Tablestore data types. The following table lists the data types.

Category	Tablestore data type
Integer	INTEGER
Floating point	DOUBLE
String	STRING
Boolean	BOOLEAN
Binary	BINARY

 **Note** Tablestore does not support DATE-type data. Applications use the LONG-type UNIX timestamp to indicate time.

Parameters

Parameter	Description	Required	Default value
endpoint	The endpoint of the Tablestore server.	Yes	None
accessId	The AccessKey ID of the account that is used to access Tablestore.	Yes	None
accessKey	The AccessKey secret of the account that is used to access Tablestore.	Yes	None

Parameter	Description	Required	Default value
instanceName	<p>The name of the Tablestore instance. The instance is an entity for you to use and manage Tablestore.</p> <p>After you activate the Tablestore service, you must create an instance in the Tablestore console before you can create and manage tables.</p> <p>Instances are the basic units that you can use to manage Tablestore resources. Access control and resource measurement for applications are implemented at the instance level.</p>	Yes	None
table	<p>The name of the source table. You can specify only one table as the source table. Multi-table synchronization is not required for Tablestore.</p>	Yes	None
column	<p>The columns that you want to synchronize from the source table. The columns are described in a JSON array. Tablestore is a NoSQL database service. You must specify column names for Tablestore Reader to read data.</p> <ul style="list-style-type: none"> You can specify common columns. For example, you can specify {"name": "col1"} for Tablestore Reader to read data from column 1. You can specify partial columns. Tablestore Reader reads data only from the specified columns. You can specify constant columns. For example, you can specify {"type": "STRING", "value": "DataX"} for Tablestore Reader to read data from the column in which data is of the STRING type and the data value is DataX. The type parameter specifies the constant type. The supported types are STRING, INT, DOUBLE, Boolean, BINARY, INF_MIN, and INF_MAX. If the constant type is BINARY, the constant value must be Base64-encoded. INF_MIN indicates the minimum value specified by Tablestore, and INF_MAX indicates the maximum value specified by Tablestore. If you set the type to INF_MIN or INF_MAX, do not set the value. If you set the value, errors may occur. You cannot specify a function or custom expression. This is because Tablestore does not provide functions or expressions that are similar to those of SQL. Tablestore Reader cannot read data from columns that contain functions or expressions. 	Yes	None

Parameter	Description	Required	Default value
begin and end	<p>The Tablestore table range from which you want to read data. You can specify both or neither of the two parameters. The begin and end parameters define the value ranges of primary key columns in the Tablestore table. Make sure that you specify the value ranges for all primary key columns in the table. If you do not need to limit a range, specify the parameters as {"type": "INF_MIN"} and {"type": "INF_MAX"}. For example, to read data from a Tablestore table with the primary key of [DeviceID, SellerID], specify the begin and end parameters in the following way:</p> <pre data-bbox="392 595 1112 1021"> "range": { "begin": [{"type": "INF_MIN"}, // The minimum value of the DeviceID field. {"type": "INT", "value": "0"} // The minimum value of the SellerID field.], "end": [{"type": "INF_MAX"}, // The maximum value of the DeviceID field. {"type": "INT", "value": "9999"} // The maximum value of the SellerID field.] } </pre> <p>To read all data from the table, specify the begin and end parameters in the following way:</p> <pre data-bbox="392 1137 1112 1509"> "range": { "begin": [{"type": "INF_MIN"}, // The minimum value of the DeviceID field. {"type": "INF_MIN"} // The minimum value of the SellerID field.], "end": [{"type": "INF_MAX"}, // The maximum value of the DeviceID field. {"type": "INF_MAX"} // The maximum value of the SellerID field.] } </pre>	Yes	None

Parameter	Description	Required	Default value
split	<p>The custom rule for data sharding. This parameter is an advanced configuration item. We recommend that you do not set this parameter.</p> <p>If data is unevenly distributed in a Tablestore table and the automatic sharding feature of Tablestore Reader fails to work, you can customize a sharding rule.</p> <p>The sharding rule that is specified by the split parameter must fall in the range that is specified by the begin and end parameters and must be the values of the partition key. This means that you specify only the values of the partition key instead of the values of primary key columns in the split parameter.</p> <p>To read data from a Tablestore table with the primary key of [DeviceID, SellerID], specify the following parameters:</p> <pre> "range": { "begin": { {"type": "INF_MIN"}, // The minimum value of the DeviceID field. {"type": "INF_MIN"} // The minimum value of the SellerID field. }, "end": { {"type": "INF_MAX"}, // The maximum value of the DeviceID field. {"type": "INF_MAX"} // The maximum value of the SellerID field. }, // The specified sharding rule. If you specify a sharding rule, the synchronization node is split into concurrent tasks based on the values of the begin, end, and split parameters. Data is sharded based only on the partition key, which is the first column of the primary key. // The data type of the partition key can be INF_MIN, INF_MAX, STRING, or INT. "split": [{"type": "STRING", "value": "1"}, {"type": "STRING", "value": "2"}, {"type": "STRING", "value": "3"}, {"type": "STRING", "value": "4"}, {"type": "STRING", "value": "5"}] } </pre>	No	None

Configure Tablestore Reader by using the codeless UI

This method is not supported.

Configure Tablestore Reader by using the code editor

In the following code, a node is configured to read data from a Tablestore table:

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {

```

```

"stepType":"ots",// The reader type.
"parameter":{
  "datasource":"","// The data source.
  "column":[// The columns from which data is read.
    {
      "name":"column1">// The name of the column.
    },
    {
      "name":"column2"
    },
    {
      "name":"column3"
    },
    {
      "name":"column4"
    },
    {
      "name":"column5"
    }
  ],
  "range":{
    "split":[
      {
        "type":"INF_MIN"
      },
      {
        "type":"STRING",
        "value":"splitPoint1"
      },
      {
        "type":"STRING",
        "value":"splitPoint2"
      },
      {
        "type":"STRING",
        "value":"splitPoint3"
      },
      {
        "type":"INF_MAX"
      }
    ],
    "end":[
      {
        "type":"INF_MAX"
      },
      {
        "type":"INF_MAX"
      },
      {
        "type":"STRING",
        "value":"end1"
      },
      {
        "type":"INT",
        "value":"100"
      }
    ],
    "begin":[

```

```
begin :
  {
    "type":"INF_MIN"
  },
  {
    "type":"INF_MIN"
  },
  {
    "type":"STRING",
    "value":"begin1"
  },
  {
    "type":"INT",
    "value":"0"
  }
]
},
"table":"","// The name of the source table.
},
"name":"Reader",
"category":"reader"
},
{
  "stepType":"stream",
  "parameter":{},
  "name":"Writer",
  "category":"writer"
}
],
"setting":{
  "errorLimit":{
    "record":"0">// The maximum number of dirty data records allowed.
  },
  "speed":{
    "throttle":false,// Specifies whether to enable bandwidth throttling. The value false indicates that bandwidth throttling is disabled, and the value true indicates that bandwidth throttling is enabled. The concurrent parameter takes effect only when the throttle parameter is set to true.
    "concurrent":1,// The maximum number of concurrent threads.
  }
},
"order":{
  "hops":[
    {
      "from":"Reader",
      "to":"Writer"
    }
  ]
}
}
```

2.4.5.3.12. Configure PostgreSQL Reader

This topic describes the data types and parameters supported by PostgreSQL Reader and how to configure it by using the codeless UI and code editor.

PostgreSQL Reader connects to a remote PostgreSQL database and runs a SELECT statement to select and read data from the database. ApsaraDB for Relational Database Service (RDS) provides the PostgreSQL storage engine.

Specifically, PostgreSQL Reader connects to a remote PostgreSQL database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and sends the statement to the database. The PostgreSQL database runs the statement and returns the result. Then, PostgreSQL Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

- PostgreSQL Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SQL statement to the PostgreSQL database.
- If you specify the querySql parameter, PostgreSQL Reader directly sends the value of this parameter to the PostgreSQL database.

Data types

PostgreSQL Reader supports most PostgreSQL data types. Ensure that your data types are supported.

The following table lists the data types supported by PostgreSQL Reader.

Category	PostgreSQL data type
Integer	bigint, bigserial, integer, smallint, and serial
Float	double, precision, money, numeric, and real
String	varchar, char, text, bit, and inet
Date and time	date, time, and timestamp
Boolean	boolean
Binary	bytea

Note

- Except for the preceding data types, other types are not supported.
- You need to convert the money, inet, and bit types by using syntax such as `a_inet::varchar`.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized.	Yes	None

Parameter	Description	Required	Default value
column	<p>An array of columns to be synchronized from the configured table, in JSON format. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported, which means that you can select and export specific columns. Change of the column order is supported, which means that you can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in compliance with SQL syntax supported by MySQL. For example, ["id", "table", "1", "mingya.wmy", "null", "to_char(a+1)", "2.3", "true"]. <ul style="list-style-type: none"> id: a column name. table: the name of a column that contains reserved keywords. 1: an integer constant. 'mingya.wmy': a string constant, which is enclosed in a pair of single quotation marks ('). 'null': a string. to_char(a + 1): a function expression. 2.3: a float value. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized. It cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when PostgreSQL Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then initiates concurrent data synchronization threads, which improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. Currently, the splitPk parameter supports data sharding only for integers but not for other data types such as string, float, and date. If you specify this parameter to a column of an unsupported type, PostgreSQL Reader ignores the splitPk parameter and synchronizes data through a single thread. If you do not specify the splitPk parameter or leave it empty, Data Integration synchronizes data through a single thread. 	No	None
where	<p>The WHERE clause. PostgreSQL Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to id>2 and sex=1 .</p> <ul style="list-style-type: none"> The WHERE clause can be used for synchronizing incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. 	No	None

Parameter	Description	Required	Default value
querySql (only available in the code editor)	The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter. For example, if you need to join multiple tables for data synchronization, set this parameter to <code>select a,b from table_a join table_b on table_a.id = table_b.id</code> . If you specify the querySql parameter, PostgreSQL Reader ignores the table, column, and where parameters that you have configured.	No	None
fetchSize	The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects reading efficiency. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note A value larger than 2048 may lead to the out of memory (OOM) error during the data synchronization process.</p> </div>	No	512

Configure PostgreSQL Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the data synchronization node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Filter	The filter condition for the data to be synchronized. Currently, filtering based on the limit keyword is not supported. The SQL syntax is determined by the selected data store.
Shard Key	The shard key. You can use a column in the source table as the shard key. We recommend that you use the primary key or an indexed column. Only integer fields are supported. If data sharding is performed based on the configured shard key, data can be read concurrently to improve data synchronization efficiency. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note The Shard Key parameter is displayed only when you configure the source connection for a data synchronization node.</p> </div>

2. Configure field mapping (the column parameter in the preceding parameter description).

Fields in the source table (left) have a one-to-one mapping with fields in the destination table (right). You can click **Add** to add a field or move the pointer over a field and click the **Delete** icon to delete a field.

Configuration item	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.

Configuration item	Description
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	The fields are automatically sorted based on specified rules.
Change Fields	You can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, while other blank rows are ignored.
Add	<ul style="list-style-type: none"> You can enter constants. Each constant must be enclosed in single quotation marks ('), such as 'abc' and '123'. You can use scheduling parameters, such as \${bizdate}. You can enter functions supported by relational databases, such as now() and count(1). If the value you entered cannot be parsed, the type is displayed as Unidentified.

3. Configure the channel.

Parameter	Description
DMU	<p>The billing unit of Data Integration.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note Use caution while setting the DMU parameter. Its value limits the maximum number of concurrent threads.</p> </div>
Concurrent Threads	The maximum number of concurrent data synchronization threads. If you specify this parameter, the data records are split based on the shard key specified for the reader so that they are synchronized in multiple threads concurrently. This improves the transmission rate.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the data synchronization node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the data synchronization node on the custom resource group. Set the resource group properly based on network conditions of the data stores, resource group usage, and business importance.

Configure PostgreSQL Reader by using the code editor

In the following code, a node is configured to read data from a PostgreSQL database.

```

{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"postgresql",// The reader type.
      "parameter":{
        "datasource":"","// The connection name.
        "column":[// The columns to be synchronized.
          "col1",
          "col2"
        ],
        "where":"","// The WHERE clause.
        "splitPk":"","// The shard key based on which the table is sharded. Data Integration initiates concurrent threads to synchronize data.
        "table":"","// The name of the table to be synchronized.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      // The following template is used to configure the writer. For more information, see the document of the corresponding writer.
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0",// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1,// The maximum number of concurrent threads.
      "dmu":1// The DMU value.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}

```

Additional instructions

- Data synchronization between primary and secondary databases

A secondary PostgreSQL database can be deployed for disaster recovery. The secondary database continuously synchronizes data from the primary database based on binlogs. Especially when network conditions are unfavorable, data latency between the primary and secondary databases is unavoidable, which can lead to data inconsistency.

- Concurrency control

PostgreSQL is a relational database management system (RDBMS), which supports strong consistency for data queries. A database snapshot is created before a data synchronization node starts. PostgreSQL Reader reads data from the database snapshot. Therefore, if new data is written to the database during data synchronization, the reader cannot obtain the new data.

Data consistency cannot be ensured when you enable PostgreSQL Reader to run concurrent threads on a single data synchronization node.

PostgreSQL Reader shards the table based on the `splitPk` parameter and runs multiple concurrent threads to synchronize data. These concurrent threads belong to different transactions, and they read data at different times. This means that the concurrent threads observe different snapshots.

Theoretically, the data inconsistency issue is unavoidable if a single data synchronization node includes multiple threads. However, two workarounds are available:

- Do not enable concurrent threads on a single data synchronization node. Essentially, do not specify the `splitPk` parameter. In this way, data consistency is ensured while data is synchronized at a low efficiency.
- Disable writers to ensure that the data is unchanged during data synchronization. For example, lock the table and disable data synchronization between primary and secondary databases. In this way, data is synchronized efficiently but your ongoing services can be interrupted.

- Character encoding

A PostgreSQL database supports only EUC_CN and UTF-8 encoding formats for simplified Chinese characters. PostgreSQL Reader uses JDBC, which can automatically convert encoding of characters. Therefore, you do not need to specify the encoding.

If data is written to the PostgreSQL database in an encoding format different from that specified by the PostgreSQL database, PostgreSQL Reader cannot recognize this inconsistency and may export garbled characters.

- Incremental data synchronization

PostgreSQL Reader connects to a database through JDBC and uses a `SELECT` statement with a `WHERE` clause to read incremental data in either of the following ways:

- For batch data, incremental add, update, and delete operations (including logical delete operations) are distinguished by timestamps. Specify the `WHERE` clause based on the timestamp. The timestamp must be later than the latest timestamp in the last synchronization.
- For streaming data, specify the `WHERE` clause based on the data record ID. The data record ID must be larger than the maximum ID involved in the last synchronization.

If incremental data cannot be distinguished, PostgreSQL Reader cannot perform incremental synchronization but can perform full synchronization only.

- Syntax validation

PostgreSQL Reader allows you to specify custom `SELECT` statements by using the `querySql` parameter but does not verify the syntax of the custom `SELECT` statements.

2.4.5.3.13. Configure SQL Server Reader

This topic describes the data types and parameters supported by SQL Server Reader and how to configure it by using the codeless user interface (UI) and code editor.

SQL Server Reader connects to a remote SQL Server database and runs a `SELECT` statement to select and read data from the database.

Specifically, SQL Server Reader connects to a remote SQL Server database through Java Database Connectivity (JDBC), generates a SELECT statement based on your configurations, and then sends the statement to the database. The SQL Server database runs the statement and returns the result. Then, SQL Server Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

- SQL Server Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SELECT statement to the SQL Server database.
- If you specify the `querySql` parameter, SQL Server Reader directly sends the value of this parameter to the SQL Server database.

SQL Server Reader supports most SQL Server data types. Make sure that your data types are supported.

The following table lists the data types supported by SQL Server Reader.

Category	SQL Server data type
Integer	bigint, int, smallint, and tinyint
Floating point	float, decimal, real, and numeric
String	char, nchar, ntext, nvarchar, text, varchar, nvarchar (max), and varchar (max)
Date and time	date, datetime, and time
Boolean	bit
Binary	binary, varbinary, varbinary (max), and timestamp

Parameters

Parameter	Description	Required	Default value
<code>datasource</code>	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
<code>table</code>	The name of the table to be synchronized. You can select only one source table for each sync node.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select and export specific columns. Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in compliance with the SQL syntax supported by MySQL. For example, ["id", "table", "1", "mingya.wmy", "null", "to_char(a+1)", "2.3", "true"] . <ul style="list-style-type: none"> id: a column name. table: the name of a column that contains reserved keywords. 1: an integer constant. 'mingya.wmy': a string constant, which is enclosed in single quotation marks (' '). 'null': a string. to_char(a + 1): a function expression. 2.3: a floating-point constant. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when SQL Server Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. Currently, the splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you specify this parameter to a column of an unsupported type, SQL Server Reader returns an error. 	No	None
where	<p>The WHERE clause. SQL Server Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to limit 10 during a test. For example, if you need to synchronize data generated on the current day, set this parameter to <code>gmt_create > \$bizdate</code> .</p> <ul style="list-style-type: none"> You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. 	No	None

Parameter	Description	Required	Default value
querySql	The SELECT statement used for refined data filtering. Specify this parameter in the following format: "querySql": "SELECT statement", . If you specify this parameter, Data Integration directly filters data based on this parameter. For example, if you want to join multiple tables for data synchronization, set this parameter to <code>select a,b from table_a join table_b on table_a.id = table_b.id</code> . If you specify the querySql parameter, SQL Server Reader ignores the table, column, and where parameters that you have configured.	No	None
fetchSize	The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects reading efficiency. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> ? Note A value larger than 2048 may lead to the out of memory (OOM) error during the data synchronization process. </div>	No	1024

Configure SQL Server Reader by using the codeless UI

1. Configure the connections.

Configure the source and destination connections for the sync node.

Configuration item	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and enter the name of a connection that has been configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Filter	The filter condition for the data to be synchronized. Currently, filtering based on the limit keyword is not supported. The SQL syntax is determined by the selected connection.
Shard Key	The shard key. You can use a column in the source table as the shard key. We recommend that you use the primary key or an indexed column.

2. Configure field mapping, that is, the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

Configuration item	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.

Configuration item	Description
Auto Layout	Click Auto Layout. The fields are automatically sorted based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box that appears, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. ◦ You can use scheduling parameters, such as \${bizdate}. ◦ You can enter functions supported by relational databases, such as now() and count(1). ◦ Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Configuration item	Description
Expected Concurrency	The maximum number of concurrent threads to read data from or write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure SQL Server Reader by using the code editor

In the following code, a node is configured to read data from an SQL Server database.

```

{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"sqlserver",// The reader type.
      "parameter":{
        "datasource":"","// The connection name.
        "column":[// The columns to be synchronized.
          "id",
          "name"
        ],
        "where":"","// The WHERE clause.
        "splitPk":"","// The shard key based on which the table is sharded.
        "table":"","// The name of the table to be synchronized.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      //The following template is used to configure the writer. For more information, see the corresponding topic.
      "stepType":"stream",
      "parameter":{
        "name":"Writer",
        "category":"writer"
      }
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0",// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1,// The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}

```

If you want to use the `querySql` parameter to specify a `SELECT` statement to query data, see the following sample code in the script of SQL Server Reader. Assume that the SQL Server connection is `sql_server_source`, the table to be queried is `dbo.test_table`, and the column to be queried is `name`.

```
{
  "stepType": "sqlserver",
  "parameter": {
    "querySql": "select name from dbo.test_table",
    "datasource": "sql_server_source",
    "column": [
      "name"
    ],
    "where": "",
    "splitPk": "id"
  },
  "name": "Reader",
  "category": "reader"
},
```

Additional instructions

- Data synchronization between primary and secondary databases

A secondary SQL Server database can be deployed for disaster recovery. The secondary database continuously synchronizes data from the primary database based on binlogs. Especially when network conditions are unfavorable, data latency between the primary and secondary databases is unavoidable, which can lead to data inconsistency.

- Concurrency control

SQL Server is a relational database management system (RDBMS), which supports strong consistency for data queries. A database snapshot is created before a sync node starts. SQL Server Reader reads data from the database snapshot. Therefore, if new data is written to the database during data synchronization, the reader cannot obtain the new data.

Data consistency cannot be guaranteed when you enable SQL Server Reader to run concurrent threads on a single sync node.

SQL Server Reader shards the table based on the splitPk parameter and runs multiple concurrent threads to synchronize data. These concurrent threads belong to different transactions. They read data at different time points. This means that the concurrent threads observe different snapshots.

Theoretically, the data inconsistency issue is unavoidable if a single sync node includes multiple threads. However, two workarounds are available:

- Do not enable concurrent threads on a single sync node. Essentially, do not specify the splitPk parameter. In this way, data consistency is guaranteed although data is synchronized at a low efficiency.
- Disable writers to make sure that the data is unchanged during data synchronization. For example, lock the table and disable data synchronization between primary and secondary databases. In this way, data is synchronized efficiently but your ongoing services may be interrupted.

- Character encoding

SQL Server Reader uses JDBC, which can automatically convert the encoding of characters. Therefore, you do not need to specify the encoding format.

- Incremental data synchronization

SQL Server Reader connects to a database through JDBC and uses a SELECT statement with a WHERE clause to read incremental data in the following ways:

- For data in batches, incremental add, update, and delete operations (including logically delete operations) are distinguished by timestamps. Specify the WHERE clause based on the timestamp. The timestamp must be later than the latest timestamp in the last synchronization.

- For streaming data, specify the WHERE clause based on the data record ID. The data record ID must be larger than the maximum ID involved in the last synchronization.

If incremental data cannot be distinguished, SQL Server Reader cannot perform incremental synchronization but can perform full synchronization only.

- Syntax validation

SQL Server Reader allows you to specify custom SELECT statements by using the querySql parameter but does not verify the syntax of the custom SELECT statements.

2.4.5.3.14. Configure LogHub Reader

This topic describes the data types and parameters supported by LogHub Reader and how to configure it by using the codeless UI and code editor.

As an all-in-one real-time data logging service, Log Service provides features to collect, consume, deliver, query, and analyze log data. It can comprehensively improve the capabilities to process and analyze numerous logs. LogHub Reader consumes real-time log data in LogHub by using the Java SDK for Log Service, converts the data to a format that can be read by the Data Integration service, and sends the converted data to a writer.

How it works

LogHub Reader consumes real-time log data in LogHub by using the following version of the Java SDK for Log Service:

```
<dependency>
  <groupId>com.aliyun.openservices</groupId>
  <artifactId>aliyun-log</artifactId>
  <version>0.6.7</version>
</dependency>
```

In Log Service, Logstore is a basic unit for collecting, storing, and querying log data. The read and write logs of a Logstore are stored in a shard. Each Logstore consists of several shards, each of which is defined by a left-closed and right-open interval of MD5 values so that intervals do not overlap each other. The range of all intervals covers all the allowed MD5 values. Each shard can independently provide some services.

- Write: 5 Mbit/s, 2,000 times/s.
- Read: 10 Mbit/s, 100 times/s.

LogHub Reader consumes log data in shards by following this process in which the GetCursor and BatchGetLog API operations are called:

- Obtain a cursor based on the time range.
- Read logs based on the cursor and step parameters and return the next cursor.
- Keep moving the cursor to consume logs.
- Split the node to concurrent threads based on shards.

Data types

The following table lists the data types supported by LogHub Reader.

Data Integration data type	LogHub data type
STRING	STRING

Parameters

Parameter	Description	Required	Default value
endpoint	The Log Service endpoint, which is a URL for accessing a project and log data. It varies based on the Alibaba Cloud region where the project resides and the project name.	Yes	None
accessId	The AccessKey ID for connecting to Log Service.	Yes	None
accessKey	The AccessKey secret for connecting to Log Service.	Yes	None
project	The name of the project. A project is the basic unit for managing resources in Log Service. You can exercise access control at the project level, and isolate resources among different projects.	Yes	None
logstore	The name of the Logstore. A Logstore is the basic unit for collecting, storing, and querying log data in Log Service.	Yes	None
batchSize	The number of entries queried from Log Service at a time.	No	128
column	<p>The column name in each log entry. You can configure a column that stores metadata in a source table of LogHub in such a way that the metadata in this column is inserted into the destination table. Supported metadata includes the log topic, unique identifier of the collection machine, host name, path, and log time.</p> <p>Note The column name is case-sensitive.</p>	Yes	None
beginDateTime	<p>The start time of data consumption. The value is the time when log data arrives at LogHub. This parameter defines the left boundary of a left-closed and right-open interval in the format of yyyyMMddHHmmss, for example, 20180111013000. The parameter can work with the scheduling time parameter in DataWorks.</p> <p>Note The beginDateTime and endDateTime parameters must be used in pairs.</p>	You must specify either beginDateTime or beginTimestampMillis, but not both.	None
endDateTime	<p>The end time of data consumption in the format of yyyyMMddHHmmss, such as 20180111013010. This parameter defines the right boundary of a left-closed and right-open interval and can work with the scheduling time parameter in DataWorks.</p> <p>Note Make sure that the time specified by the endDateTime parameter of the previous interval is the same as or later than the time specified by the beginDateTime parameter of the current interval. If the intervals do not overlap, data may fail to be read in some regions.</p>	You must specify either endDateTime or endTimestampMillis, but not both.	None

Parameter	Description	Required	Default value
beginTimestampMillis	<p>The start time of data consumption. This parameter specifies the left boundary of the left-closed and right-open interval, measured in milliseconds.</p> <p>Note The beginTimestampMillis and endTimestampMillis parameters must be used in pairs.</p> <p>A value of -1 indicates the position where the cursor starts in Log Service, which is specified by CursorMode.BEGIN. We recommend that you specify the beginDateTime parameter.</p>	You must specify either beginTimestampMillis or beginDateTime, but not both.	None
endTimestampMillis	<p>The end time of data consumption, measured in milliseconds. This parameter defines the right boundary of the left-closed and right-open interval.</p> <p>Note The endTimestampMillis and beginTimestampMillis parameters must be used in pairs.</p> <p>A value of -1 indicates the position where the cursor ends in Log Service, which is specified by CursorMode.END. We recommend that you specify the endDateTime parameter.</p>	You must specify either endTimestampMillis or endDateTime, but not both.	None

Configure LogHub Reader in the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The type and name of the source connection. Select a connection that you have configured in DataWorks.
Logstore	The name of the Logstore from which data is read.
Start Timestamp	The start time of data consumption. The value is the time when log data arrives at LogHub. This parameter defines the left boundary of a left-closed and right-open interval in the format of yyyyMMddHHmmss, for example, 20180111013000. The parameter can work with the scheduling time parameter in DataWorks.
End Timestamp	The end time of data consumption in the format of yyyyMMddHHmmss, such as 20180111013010. This parameter defines the right boundary of a left-closed and right-open interval and can work with the scheduling time parameter in DataWorks.
Records per Batch	The number of entries queried from Log Service at a time.

2. Configure field mapping. It is equivalent to setting the column parameter provided in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), such as 'abc' and '123'. ◦ You can use scheduling parameters, such as \${bizdate}. ◦ You can enter functions supported by relational databases, such as now() and count(1). ◦ Fields that cannot be parsed are indicated by Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure LogHub Reader by using the code editor

In the following code, a node is configured to read data from LogHub. For more information about the parameters, see the preceding parameter description.

```

{
  "type": "job",
  "version": "2.0" // The version number.
  "steps": [
    {
      "stepType": "loghub" // The reader type.
      "parameter": {
        "datasource": "" // The connection name.
        "column": [ // The columns to be synchronized.
          "col0",
          "col1",
          "col2",
          "col3",
          "col4",
          "=Topic" // The log topic.
          "HostName" // The hostname.
          "Path" // The path.
          "LogTime" // The log time.
        ],
        "beginDateTime": "" // The start time of data consumption.
        "batchSize": "" // The number of entries that are queried from Log Service at a time.
        "endDateTime": "" // The end time of data consumption.
        "fieldDelimiter": "," // The column delimiter.
        "encoding": "UTF-8" // The encoding format.
        "logstore": "" // The name of the target Logstore.
      },
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent": 1 // The maximum number of concurrent threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

 **Note** If the metadata in JSON format is prefixed by tag, delete the tag prefix. For example, change `__tag__:__client_ip__` to `__client_ip__`.

2.4.5.3.15. Configure Tablestore Reader-Internal

This topic describes the data types and parameters supported by Tablestore Reader-Internal and how to configure it by using the code editor.

Tablestore is a NoSQL database service built on the Apsara distributed operating system that allows you to store and access large amounts of structured data in real time. Tablestore organizes data into instances and tables. It can seamlessly expand the data scale by using data sharding and load balancing technologies.

Tablestore Reader-Internal is used to export data for the Tablestore Internal model, whereas Tablestore Reader is used to export data for the Tablestore Public model.

Tablestore Reader-Internal can export data in multi-version mode or normal mode:

- **Multi-version mode:** Tablestore stores multiple versions of column values, and this mode allows you to export data of multiple versions.

Tablestore Reader-Internal converts a cell to a 4-tuple of a one-dimensional table: PrimaryKey (columns 1 to 4), ColumnName, Timestamp, and Value. This process is similar to that for the multi-version mode of HBase Reader. Each {PrimaryKey, ColumnName, Timestamp, Value} tuple is sent to a writer as four columns in Data Integration records.

- **Normal mode:** This mode allows you to export the latest version of each column in each row, which is the same as the normal mode of HBase Reader. For more information, see the normal mode of HBase Reader in [Configure an HBase connection](#).

Tablestore Reader-Internal connects to a Tablestore server by using the official Java SDK for Tablestore and reads data from the server. Tablestore Reader-Internal optimizes the read process by providing features such as performing retry attempts when a timeout or exception occurs.

Tablestore Reader-Internal supports all Tablestore data types. The following table lists the data types supported by Tablestore Reader-Internal.

Data Integration data type	Tablestore data type
LONG	INTEGER
DOUBLE	DOUBLE
STRING	STRING
BOOLEAN	BOOLEAN
BYTES	BINARY

Parameters

Parameter	Description	Required	Default value
mode	The mode in which Tablestore Reader-Internal exports data. Valid values: <i>normal</i> and <i>multiVersion</i> .	Yes	None
endpoint	The endpoint of the Tablestore server.	Yes	None

Parameter	Description	Required	Default value
accessId	The AccessKey ID for connecting to Tablestore.	Yes	None
accessKey	The AccessKey secret for connecting to Tablestore.	Yes	None
instanceName	<p>The name of the Tablestore instance. The instance is an entity for you to use and manage Tablestore.</p> <p>After you activate the Tablestore service, you must create an instance in the console before you create and manage tables. Instances are the basic units for managing Tablestore resources. All access control and resource measurement for applications are implemented at the instance level.</p>	Yes	None
table	The name of the source table. You can specify only one table as the source table. Multi-table synchronization is not required for Tablestore.	Yes	None
range	<p>The range of the data to export, in the format of [begin,end).</p> <ul style="list-style-type: none"> If the value of the begin parameter is smaller than that of the end parameter, data is read in forward order. If the value of the begin parameter is larger than that of the end parameter, data is read in reverse order. The value of the begin parameter cannot be the same as that of the end parameter. The following value types are supported: STRING, INT, and BINARY. Binary data is passed in as Base64 strings in binary format. INF_MIN represents an infinitely small value and INF_MAX represents an infinitely large value. 	No	By default, data is read from the beginning of the table to the end of the table.
range: {"begin"}	<p>The start of the data to export. Enter an empty array, a primary key prefix, or a complete primary key. In forward order, the default primary key suffix is INF_MIN. In reverse order, the default primary key suffix is INF_MAX.</p> <p>This parameter specifies the value range of the Tablestore primary key and is used for data filtering. If you do not specify this parameter, the minimum value is used by default.</p> <p>The JSON format does not support binary data. If the data type of the PrimaryKey column is BINARY, you must use the Java method Base64.encodeBase64String to convert binary data to a string, and then enter the string as the value of the parameter. Example:</p> <ul style="list-style-type: none"> <code>byte[] bytes = "hello".getBytes();</code> : constructs binary data, which is the byte value of the string hello. <code>String inputValue = Base64.encodeBase64String(bytes)</code> : calls the Base64.encodeBase64String method to convert the binary data to a string. <p>After you run the preceding code, the string "aGVsbG8=" is returned for the inputValue parameter.</p> <p>Finally, set this parameter to <code>{"type":"binary","value":"aGVsbG8="}</code> .</p>	No	Data is read from the beginning of the table.

Parameter	Description	Required	Default value
range: {"end" }	<p>The end of the data to export. Enter an empty array, a primary key prefix, or a complete primary key. In forward order, the default primary key suffix is INF_MAX. In reverse order, the default primary key suffix is INF_MIN.</p> <p>The JSON format does not support binary data. If the data type of the PrimaryKey column is BINARY, you must use the Java method Base64.encodeBase64String to convert binary data to a string, and then enter the string as the value of the parameter. Example:</p> <ul style="list-style-type: none"> <code>byte[] bytes = "hello".getBytes();</code> : constructs binary data, which is the byte value of the string hello. <code>String inputValue = Base64.encodeBase64String(bytes)</code> : calls the Base64.encodeBase64String method to convert the binary data to a string. <p>After you run the preceding code, the string "aGVsbG8=" is returned for the inputValue parameter.</p> <p>Finally, set this parameter to <code>{"type":"binary", "value":"aGVsbG8="}</code>.</p>	No	Data is read until the end of the table.
range:{"split" }	<p>If an excessively large amount of data needs to be exported, you can specify this parameter to split one node to multiple concurrent threads.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> The field based on which the node is split must be the shard key, which is the first column of the primary key, and the data type of the field must be the same as that of the partition key. The specified field must be within the value range that is specified by the begin and end parameters. The values of this field must be sorted in the descending or ascending order based on the data reading order that is determined by values of the begin and end parameters. </div>	No	No sharding rule is specified.

Parameter	Description	Required	Default value
column	<p>The columns to be exported. Both regular and constant columns can be exported. A regular column is in the format of <code>{"name":"your column name"}</code>.</p> <div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> Constant columns cannot be exported in multi-version mode. You cannot specify the PrimaryKey column. The exported tuple data contains the complete primary key by default. Each column can be exported only once. </div>	None	All versions of all columns are exported.
timeRange (applicable only to the multi-version mode)	<p>The time range of the requested data, in the format of [begin,end].</p> <div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p>Note The value of the begin parameter must be smaller than that of the end parameter.</p> </div>	No	All the data is read.
timeRange: {"begin"} (applicable only to the multi-version mode)	The start time for reading data. Valid values: 0 to LONG_MAX.	No	0
timeRange: {"end"} (applicable only to the multi-version mode)	The end time for reading data. Valid values: 0 to LONG_MAX.	No	<i>LONG_MAX (9223372036854775806L)</i>
maxVersion (applicable only to the multi-version mode)	The specified version of the requested data. Valid values: 1 to INT32_MAX.	No	The data of all versions is read.

Configure Tablestore Reader-Internal by using the codeless UI

The codeless UI is not supported for Tablestore Reader-Internal.

Configure Tablestore Reader-Internal by using the code editor

- Multi-version mode

```
{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "reader": {
      "plugin": "otsreader-internalreader",
      "parameter": {
        "mode": "multiVersion"
```

```
mode : multiversion ,
"endpoint": "",
"accessId": "",
"accessKey": "",
"instanceName": "",
"table": "",
"range": {
  "begin": [
    {
      "type": "string",
      "value": "a"
    },
    {
      "type": "INF_MIN"
    }
  ],
  "end": [
    {
      "type": "string",
      "value": "g"
    },
    {
      "type": "INF_MAX"
    }
  ],
  "split": [
    {
      "type": "string",
      "value": "b"
    },
    {
      "type": "string",
      "value": "c"
    }
  ]
},
"column": [
  {
    "name": "attr1"
  }
],
"timeRange": {
  "begin": 1400000000,
  "end": 1600000000
},
"maxVersion": 10
}
},
"writer": {}
}
```

- Normal mode

```
{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "mode": "normal"
  }
}
```

```
"reader": {
  "plugin": "otsreader-internalreader",
  "parameter": {
    "mode": "normal",
    "endpoint": "",
    "accessId": "",
    "accessKey": "",
    "instanceName": "",
    "table": "",
    "range": {
      "begin": [
        {
          "type": "string",
          "value": "a"
        },
        {
          "type": "INF_MIN"
        }
      ],
      "end": [
        {
          "type": "string",
          "value": "g"
        },
        {
          "type": "INF_MAX"
        }
      ],
      "split": [
        {
          "type": "string",
          "value": "b"
        },
        {
          "type": "string",
          "value": "c"
        }
      ]
    }
  },
  "column": [
    {
      "name": "pk1"
    },
    {
      "name": "pk2"
    },
    {
      "name": "attr1"
    },
    {
      "type": "string",
      "value": ""
    },
    {
      "type": "int",
      "value": ""
    },
    {
```

```
        "type": "double",  
        "value": ""  
    },  
    {  
        "type": "binary",  
        "value": "aGVsbG8="  
    }  
  ]  
}  
},  
"writer": {}  
}
```

2.4.5.3.16. Configure OTSStream Reader

This topic describes the data types and parameters supported by OTSStream Reader and how to configure it by using the code editor.

OTSStream Reader is mainly used to synchronize the incremental data of Tablestore. Incremental data can be considered as operation logs that include data and operation information.

Unlike plug-ins used to synchronize full data, OTSStream Reader supports only the multi-version mode. When you use OTSStream Reader to synchronize incremental data, you cannot synchronize the data of specified columns, which is related to the principle of synchronizing incremental data. The following section describes the implementation process.

Before you use OTSStream Reader, make sure that the Stream feature is enabled for your table. You can enable this feature when you create the table, or you can use the UpdateTable operation in the SDK to enable this feature.

The following example describes how to enable the Stream feature:

```
SyncClient client = new SyncClient("", "", "", "");  
Enable this feature when you create a table.  
CreateTableRequest createTableRequest = new CreateTableRequest(tableMeta);  
createTableRequest.setStreamSpecification(new StreamSpecification(true, 24)); // The value 24 indicates that the i  
ncremental data is retained for 24 hours.  
client.createTable(createTableRequest);  
If this feature is not enabled when you create a table, enable it by using the UpdateTable operation.  
UpdateTableRequest updateTableRequest = new UpdateTableRequest("tableName");  
updateTableRequest.setStreamSpecification(new StreamSpecification(true, 24));  
client.updateTable(updateTableRequest);
```

How it works

You can enable the Stream feature and set the expiration time by using the UpdateTable operation in the SDK. After the Stream feature is enabled, the Tablestore server additionally saves your operation logs. Each partition has a sequential operation log queue. Each operation log is recycled after your specified expiration time.

The Tablestore SDK provides several Stream APIs that are used to read these operation logs. OTSStream Reader obtains incremental data by using these APIs, transforms the incremental data into multiple six-tuples (pk, colName, version, colValue, opType, and sequenceInfo), and then synchronizes them into MaxCompute.

Format of the synchronized data

In the multi-version model of Tablestore, table data is organized in a three-level mode: row, column, and version. One row can have multiple columns, and the column name is not fixed. Each column can have multiple versions, and each version has a specific timestamp (the version number).

You can perform read or write operations by using Tablestore APIs. Tablestore records the incremental data by recording the recent write and modification operations on table data. Therefore, incremental data can be considered as a set of operation records.

Tablestore supports the following three types of modification operations:

- **PutRow**: writes a row. If the row already exists, it is overwritten.
- **UpdateRow**: updates a row without the need to change other data of the original row. You can add column values, overwrite column values if the related version of the column already exists, delete all the versions of a column, or delete a version of a column.
- **DeleteRow**: deletes a row.

Tablestore generates incremental data records based on each type of operation. OTSStream Reader reads these records and synchronizes the data in the format supported by DataX.

Tablestore supports dynamic columns and the multi-version mode. Therefore, a row exported by OTSStream Reader corresponds to a version of a column rather than a row in Tablestore. A row in Tablestore may correspond to multiple synchronized rows. Each synchronized row includes the primary key value, column name, timestamp of the version for the column (version number), value of the version, and operation type. If the `isExportSequenceInfo` parameter is set to true, time series information is also included.

When the data is transformed into the format supported by DataX, the following four types of operations are defined:

- **U (UPDATE)**: writes a version of a column.
- **DO (DELETE_ONE_VERSION)**: deletes a version of a column.
- **DA (DELETE_ALL_VERSION)**: deletes all the versions of a column. Delete all the versions of the column based on the primary key and the column name.
- **DR (DELETE_ROW)**: deletes a row. Delete all the data of the row based on the primary key.

In the following example, the table has two primary key columns: `pkName1` and `pkName2`.

pkName1	pkName2	columnName	timestamp	columnValue	opType
pk1_V1	pk2_V1	col_a	1441803688001	col_val1	U
pk1_V1	pk2_V1	col_a	1441803688002	col_val2	U
pk1_V1	pk2_V1	col_b	1441803688003	col_val3	U
pk1_V2	pk2_V2	col_a	1441803688000	-	DO
pk1_V2	pk2_V2	col_b	-	-	DA
pk1_V3	pk2_V3	-	-	-	DR
pk1_V3	pk2_V3	col_a	1441803688005	col_val1	U

In this example, seven rows are synchronized, which corresponds to three rows in the Tablestore table. The primary keys for the three rows are (pk1_V1, pk2_V1), (pk1_V2, pk2_V2), and (pk1_V3, pk2_V3).

- For the row whose primary key is (pk1_V1, pk2_V1), three operations are included: writing two versions of column `col_a` and one version of column `col_b`.
- For the row whose primary key is (pk1_V2, pk2_V2), two operations are included: deleting one version of column `col_a` and all the versions of column `col_b`.

- For the row whose primary key is (pk1_V3, pk2_V3), two operations are included: deleting the row and writing one version of column col_a.

Data types

OTSStream Reader supports all Tablestore data types. The following table lists the data types supported by OTSStream Reader.

Category	OTSStream data type
Integer	INTEGER
Floating point	DOUBLE
String	STRING
Boolean	BOOLEAN
Binary	BINARY

Parameters

Parameter	Description	Required	Default value
dataSource	The name of the data source. It must be the same as the added data source. You can add data sources by using the code editor.	Yes	No default value
dataTable	The name of the table from which incremental data is synchronized. You must enable the Stream feature for a table when you create the table, or you can call the UpdateTable operation to enable this feature.	Yes	No default value

Parameter	Description	Required	Default value
statusTable	<p>The name of the table that OTSStream Reader uses to store status records. These records help find the data that is not required and improve synchronization efficiency. A status table is used to store status records. If no such table exists, OTSStream Reader automatically creates one. After the running of an offline export task is completed, you do not need to delete the table. The status records in the table can be used for the next export task.</p> <ul style="list-style-type: none"> You need only to provide a table name rather than manually creating a status table. OTSStreamReader attempts to create a status table under your instance. If no such table exists, OTSStream Reader automatically creates one. If the table already exists, OTSStream Reader determines whether the metadata of the table meets the expectation. If not, an error is reported. After the running of an export task is completed, you do not need to delete the table. The status records in the table can be used for the next export task. The table enables Time To Live (TTL), and data automatically expires, which indicates that the data volume is small. You can use the same status table to store the status records of the multiple tables that are specified by the dataTable parameter and managed by the same instance. The status records are independent of each other. <p>In conclusion, you can configure a name similar to TableStoreStreamReaderStatusTable. You must make sure that the name is inconsistent with that of a business-related table.</p>	Yes	No default value
startTimeMillis	<p>The start time (included) of the incremental data, in milliseconds.</p> <ul style="list-style-type: none"> OTSStream Reader finds a point that corresponds to the time specified by the startTimeMillis parameter from the status table, and starts to read and synchronize data from this point. If OTSStream Reader cannot find the required point, it starts to read incremental data retained by the system from the first entry, and skips the data which is written later than the time specified by startTimeMillis. 	No	No default value
endTimeMillis	<p>The end time (excluded) of the incremental data, in milliseconds.</p> <ul style="list-style-type: none"> OTSStream Reader exports data from the time specified by the startTimeMillis parameter and stops exporting data when the timestamp of a data record is later than or equal to the time specified by the endTimeMillis parameter. After all the incremental data is read, OTSStream Reader stops reading data even before the time specified by the endTimeMillis parameter. 	No	No default value
date	<p>The date on which data is synchronized. Specify this parameter in the yyyyMMdd format, such as 20151111. You must specify either the date parameter or the startTimeMillis and endTimeMillis parameters. For example, Alibaba Cloud Data Process Center performs scheduling only at the day level. Therefore, the date parameter is provided.</p>	No	No default value

Parameter	Description	Required	Default value
isExportSequenceInfo	Specifies whether to synchronize time series information which includes the time when data is written. The default value is <i>false</i> , which indicates that time series information is not synchronized.	No	No default value
maxRetries	The maximum number of retries for each request of reading incremental data from Tablestore. The default value is 30. Retries are performed at specific intervals. The total time of 30 retries is about 5 minutes. You can keep the default settings.	No	No default value
startTimeString	The start time (included) of the incremental data. in milliseconds. Specify this parameter in the <code>yyyymmddhh24miss</code> format.	No	No default value
endTimeString	The end time (excluded) of the incremental data. in milliseconds. Specify this parameter in the <code>yyyymmddhh24miss</code> format.	No	No default value
mode	The synchronization mode. If this parameter is set to <code>single_version_and_update_only</code> , data is exported by row. By default, this parameter is not specified, and data is not synchronized by column.	No	No default value

Configure OTSStream Reader by using the codeless UI

This method is not supported.

Configure OTSStream Reader by using the code editor

The following example shows how to configure a synchronization node to read the incremental data of Tablestore. For more information about the parameters, see the preceding parameter description.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "otsstream", // The reader type.
      "parameter": {
        "statusTable": "TableStoreStreamReaderStatusTable", // The name of the table that OTSStream Reader uses to store status records.
        "maxRetries": 30, // The maximum number of retries on each request of reading incremental data from Tablesore. It is set to 30 by default.
        "isExportSequenceInfo": false, // Specifies whether to synchronize the time series information.
        "datasource": "${srcDatasource}", // The name of the data source.
        "startTimeString": "${startTime}", // The start time (included) of the incremental data.
        "table": "", // The name of the table from which you want to read data.
        "endTimeString": "${endTime}" // The end time (excluded) of the incremental data.
      },
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. The value false indicates that bandwidth throttling is disabled, and the value true indicates that bandwidth throttling is enabled. The mbps parameter takes effect only when the throttle parameter is set to true.
      "concurrent": 1 // The maximum number of parallel threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.3.17. Configure RDBMS Reader

This topic describes the data types and parameters supported by RDBMS Reader and how to configure it by using the code editor.

Background information

RDBMS Reader allows you to read data from an RDBMS database. RDBMS Reader connects to a remote RDBMS database and runs a SELECT statement to select and read data from the database. RDBMS Reader can read data from databases such as Dameng, Db2, PPAS, and Sybase databases. If you need RDBMS Reader to read data from a common relational database, register the driver for the corresponding database type.

RDBMS Reader connects to a remote RDBMS database by using JDBC, generates a SELECT statement based on your configurations, and then sends the statement to the database. The RDBMS database runs the statement and returns the result. Then, RDBMS Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

- RDBMS Reader generates the SQL statement based on the table, column, and where parameters that you have configured, and sends the generated SQL statement to the RDBMS database.
- If you specify the querySql parameter, RDBMS Reader directly sends the value of this parameter to the RDBMS database.

RDBMS Reader supports most data types of a common relational database, such as numbers and characters. Make sure that your data types are supported.

Parameters

Parameter	Description	Required	Default value
	<p>The JDBC URL for connecting to the RDBMS database. The format must be in accordance with the official RDBMS specifications. You can also specify the information of the attachment facility. The format varies based on the database type. Data Integration selects an appropriate driver for data reading based on the format.</p> <ul style="list-style-type: none"> • Format for DM databases: <code>jdbc:dm://ip:port/database</code> • Format for Db2 databases: <code>jdbc:db2://ip:port/database</code> • Format for PPAS databases: <code>jdbc:edb://ip:port/database</code> <p>You can enable RDBMS Reader to support a new database by using the following method:</p> <ul style="list-style-type: none"> • Go to the RDBMS Reader directory. In the directory, <code>#{DATA_HOME}</code> indicates the main directory of Data Integration. • Open the <code>plugin.json</code> file in the RDBMS Reader directory, and add the driver of your database to the <code>drivers</code> array in the file. RDBMS Reader dynamically selects the appropriate database driver to connect to the database when nodes are run. 		

Parameter	Description	Required	Default value
jdbcUrl	<pre> { "name": "rdbmsreader", "class": "com.alibaba.datax.plugin.reader.rdbmsreader.RdbmsReader", "description": "useScene: prod. mechanism: Jdbc connection using the database, execute select sql, retrieve data from the ResultSet. warn: The more you know about the database, the less problems you encounter.", "developer": "alibaba", "drivers": ["dm.jdbc.driver.DmDriver", "com.ibm.db2.jcc.DB2Driver", "com.sybase.jdbc3.jdbc.SybDriver", "com.edb.Driver"] } ` ` ` - Add the driver package to the libs directory in the RDBMS Reader directory. ` ` ` \$tree . -- libs -- Dm7JdbcDriver16.jar -- commons-collections-3.0.jar -- commons-io-2.4.jar -- commons-lang3-3.3.2.jar -- commons-math3-3.1.1.jar -- datax-common-0.0.1-SNAPSHOT.jar -- datax-service-face-1.0.23-20160120.024328-1.jar -- db2jcc4.jar -- druid-1.0.15.jar -- edb-jdbc16.jar -- fastjson-1.1.46.sec01.jar -- guava-r05.jar -- hamcrest-core-1.3.jar -- jconn3-1.0.0-SNAPSHOT.jar -- logback-classic-1.0.13.jar -- logback-core-1.0.13.jar -- plugin-rdbms-util-0.0.1-SNAPSHOT.jar `-- slf4j-api-1.7.10.jar -- plugin.json -- plugin_job_template.json `-- rdbmsreader-0.0.1-SNAPSHOT.jar </pre>	Yes	None
username	The username for connecting to the database.	Yes	None
password	The password for connecting to the database.	Yes	None
table	The name of the source table.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> Column pruning is supported. You can select specific columns to export. The column order can be changed. You can export the specified columns in an order different from that specified in the schema of the table. Constants are supported. The column names must be arranged in JSON format. For example, ["id","1","bazhen.csy","null","to_char(a + 1)","2.3","true"] . <ul style="list-style-type: none"> id: a column name. 1: an integer constant. 'bazhen.csy': a string constant. null: a null pointer. to_char(a + 1): a function expression. 2.3: a floating-point constant. true: a Boolean value. The column parameter must explicitly specify a set of columns to be synchronized, and cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when RDBMS Reader reads data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to specific shards. The splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you specify this parameter to a column of an unsupported type, RDBMS Reader returns an error. If you do not specify the splitPk parameter or leave it empty, RDBMS Reader synchronizes data by using a single thread. 	No	An empty string
where	<p>The WHERE clause. RDBMS Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data. For example, set this parameter to limit 10.</p> <p>To synchronize data generated on the current day, set the where parameter to <code>gmt_create > \$bizdate</code> .</p> <ul style="list-style-type: none"> You can use the WHERE clause to read incremental data. If you do not specify the where parameter or leave it empty, all data is read. 	No	None

Parameter	Description	Required	Default value
querySql	<p>The SELECT statement used to for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter.</p> <p>For example, if you need to join multiple tables for data synchronization, set this parameter to <code>select a,b from table_a join table_b on table_a.id = table_b.id</code> . If you specify the querySql parameter, RDBMS Reader ignores the table, column, and where parameters that you have configured.</p>	No	None
fetchSize	<p>The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects reading efficiency.</p> <p> Note A value greater than 2048 may lead to OOM during the data synchronization process.</p>	No	1,024

Configure RDBMS Reader by using the codeless UI

The codeless UI is not supported for RDBMS Reader.

Configure RDBMS Reader by using the code editor

In the following code, a node is configured to read data from an RDBMS database.

```
{
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  },
  "setting": {
    "errorLimit": {
      "record": "0"
    },
    "speed": {
      "concurrent": 1,
      "throttle": false
    }
  },
  "steps": [
    {
      "category": "reader",
      "name": "Reader",
      "parameter": {
        "connection": [
          {
            "jdbcUrl": [
              "jdbc:dm://ip:port/database"
            ],
            "table": [
              "table"
            ]
          }
        ],
        "username": "username",
        "password": "password",
        "table": "table",
        "column": [
          "*"
        ],
        "preSql": [
          "delete from XXX;"
        ]
      },
      "stepType": "rdbms"
    },
    {
      "category": "writer",
      "name": "Writer",
      "parameter": {},
      "stepType": "stream"
    }
  ],
  "type": "job",
  "version": "2.0"
}
```

2.4.5.3.18. Configure Stream Reader

This topic describes the data types and parameters supported by Stream Reader and how to configure it by using the code editor.

Stream Reader automatically generates data from the memory. It is mainly used for performance testing for data synchronization and basic functional testing.

The following table lists the data types supported by Stream Reader.

Data type	Description
String	A sequence of characters.
Long	A long integer.
Date	A value that represents dates.
Boolean	A Boolean data type that has one of two possible values.
Bytes	An 8-bit signed two's complement integer.

Parameters

Parameter	Description	Required	Default value
column	<p>The column data and type of the source data. Multiple columns can be configured. You can set to generate random strings and specify the range. The example is as follows:</p> <pre>"column": [{ "random": "8,15" }, { "random": "10,10" }]</pre> <p>The parameters are described as follows:</p> <ul style="list-style-type: none"> "random": "8, 15": generates a random string that is 8 to 15 bytes in length. "random": "10, 10": generates a 10-byte random string. 	Yes	None
sliceRecordCount	The number of columns generated repeatedly.	Yes	None

Configure Stream Reader by using the codeless UI

Currently, the codeless user interface (UI) is not supported for Stream Reader.

Configure Stream Reader by using the code editor

In the following code, a node is configured to read data from the memory and then write the data to Stream Reader.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",// The reader type.
      "parameter":{
        "column":[// The columns to be synchronized.
          {
            "type":"string",// The data type.
            "value":"field" // The value.
          },
          {
            "type":"long",
            "value":100
          },
          {
            "dateFormat":"yyyy-MM-dd HH:mm:ss",// The format of the time.
            "type":"date",
            "value":"2014-12-12 12:12:12"
          },
          {
            "type":"bool",
            "value":true
          },
          {
            "type":"bytes",
            "value":"byte string"
          }
        ],
        "sliceRecordCount":"100000",// The number of columns repeatedly generated.
      },
      "name":"Reader",
      "category":"reader"
    },
    // The following template is used to configure the writer. For more information, see the corresponding topic.
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0",// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1,// The maximum number of concurrent threads.
    }
  },
  "order":{
    "steps":[
```

```

    "ops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.3.19. Configure Hive Reader

Parameters

Parameter	Description	Required	Default value
column	The fields to read. Example: "column": ["id", "name"] .	Yes	None
table	The name of the Hive table to read. The name is case sensitive.	Yes	None
partition	The partition information of the table to read. The last-level partition must be specified. For example, if you want to read data from a three-level partition table, set this parameter to a value that contains the last-level partition information, such as pt=20150101/type=1/biz=2.	Yes	None

Configure Hive Reader by using the code editor

In the following code, a node is configured to read data from a Hive data store.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Writer",
      "category": "writer"
    },
    {
      "stepType": "hive", // The reader type. The name is the same as that in MaxCompute.
      "parameter": {
        "parameter": {
          "column": [ // The columns to be synchronized.
            "id",
            "name"
          ],
          "table": "student_tmp_2", // The name of the table to be synchronized.
          "partition": "academy=yx/class=001", // The partition settings.
          "datasource": "hive_demo"
        }
      },
      "name": "Reader",
      "category": "reader"
    }
  ],
  "setting": {},
  "order": {
    "hops": [ // Synchronize data from the reader to the writer.
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}
```

2.4.5.3.20. Configure Elasticsearch Reader

This topic describes the working principles, features, and parameters of Elasticsearch Reader.

Working principles

- Elasticsearch Reader reads data from Elasticsearch by slicing scroll queries. The slices are processed by multiple threads of a data synchronization node.
- Data types are converted based on the mapping configuration of Elasticsearch.

Basic settings

```

{
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  },
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "jvmOption":"","
    "speed":{
      "concurrent":3,
      "throttle":false
    }
  },
  "steps":[
    {
      "category":"reader",
      "name":"Reader",
      "parameter":{
        "column":[ // The fields to read.
          "id",
          "name"
        ],
        "endpoint":""," // The endpoint.
        "index":""," // The index name.
        "password":""," // The password.
        "scroll":""," // The scroll ID.
        "search":""," // The search criteria. The value is the same as the Elasticsearch query that uses the _search API
      },
      "type":"default",
      "username":"" // The username.
    },
    "stepType":"elasticsearch"
  ],
  {
    "category":"writer",
    "name":"Writer",
    "parameter":{},
    "stepType":"stream"
  }
],
  "type":"job",
  "version":"2.0" // The version number.
}

```

Advanced features

- Supports storing all data of an Elasticsearch document in one column.
You can create a column to store all data of an Elasticsearch document.
- Supports converting semi-structured data to structured data.

Item	Description
Background	Data in Elasticsearch is deeply nested. Elasticsearch may contain fields of various types and lengths and may use Chinese names. To facilitate data computing and storage in downstream businesses, Elasticsearch Reader supports converting semi-structured data to structured data.
Principle	Elasticsearch Reader flattens nested JSON data obtained from Elasticsearch to single-dimensional data based on the paths of properties in the JSON data. Then, Elasticsearch Reader maps the single-dimensional data to structured tables. In this way, Elasticsearch data in a complex structure is converted to multiple structured tables.
Solution	<ul style="list-style-type: none"> ◦ Elasticsearch Reader converts nested JSON data to single-dimensional data by using the following path formats: <ul style="list-style-type: none"> ▪ Property ▪ Property.Child property ▪ Property[0].Child property ◦ If a property has multiple child properties, Elasticsearch Reader traverses all data of the property and splits the data to multiple tables or multiple rows in the following format: Property[*].Child property ◦ Elasticsearch Reader merges data in a string array to one property in the following format and removes duplicates: Property[] where duplicates are removed ◦ Elasticsearch Reader merges multiple properties to one property in the following format: Property 1,Property 2 ◦ Elasticsearch Reader presents optional properties in the following format: Property 1 Property 2

Parameters

Parameter	Description	Required	Default value
endpoint	The endpoint of Elasticsearch.	Yes	None
username	The username for HTTP authentication.	No	Empty string
password	The password for HTTP authentication.	No	Empty string
index	The index name in Elasticsearch.	Yes	None
type	The type name in the index of Elasticsearch.	No	Index name
pageSize	The number of data records to read at a time.	No	100

Parameter	Description	Required	Default value
search	The query parameter of Elasticsearch.	Yes	None
scroll	The scroll parameter of Elasticsearch, which sets the timestamp of the snapshot taken for a scroll.	Yes	None
sort	The field based on which the returned results are sorted.	No	None
retryCount	The number of retries after a failure.	No	<i>300</i>
connTimeOut	The connection timeout of the client.	No	<i>600,000</i>
readTimeOut	The data reading timeout of the client.	No	<i>600,000</i>
multiThread	Specifies whether to use multiple threads for an HTTP request.	No	<i>true</i>
column	The fields to read.	Yes	None
full	Specifies whether to create a column to record all data of an Elasticsearch document.	No	<i>false</i>
multi	Specifies whether to split an array to multiple rows. If you enable this feature, you need to specify additional settings.	No	<i>false</i>

Additional settings:

```
"full":false,
  "multi": {
    "multi": true,
    "key":"crn_list[*]"
  }
```

2.4.5.3.21. Configure Vertica Reader

Vertica is a column-oriented database using the Massively Parallel Processing (MPP) architecture. Vertica Reader allows you to read data from Vertica. This topic describes how Vertica Reader works, the supported parameter, and how to configure it by using the code editor.

How it works

Vertica Reader connects to a remote Vertica database by using JDBC and executes a SELECT statement to select and read data from the database.

Vertica Reader connects to a remote Vertica database by using JDBC, generates a SELECT statement based on your configurations, and then sends the statement to the database. The Vertica database executes the statement and returns the result. Then, Vertica Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and sends the datasets to a writer.

- Vertica Reader generates the SELECT statement based on the table, column, and where parameters that you have configured, and sends the generated SELECT statement to the Vertica database.
- If you specify the querySql parameter, Vertica Reader directly sends the value of this parameter to the Vertica database.

Vertica Reader accesses a Vertica database by using the Vertica database driver. Confirm the compatibility between the driver version and your Vertica database. Vertica Reader uses the following version of the Vertica database driver:

```
<dependency>
  <groupId>com.vertica</groupId>
  <artifactId>vertica-jdbc</artifactId>
  <version>7.1.2</version>
</dependency>
```

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
jdbcUrl	<p>The JDBC URL for connecting to the Vertica database. You can specify multiple JDBC URLs for a database. The JDBC URLs are described in a JSON array.</p> <p>If you specify multiple JDBC URLs, Vertica Reader verifies the connectivity of the URLs in sequence to find a valid URL. If no URL is valid, Vertica Reader returns an error.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note The jdbcUrl parameter must be included in the connection parameter.</p> </div> <p>The value of the jdbcUrl parameter must be in compliance with the standard format supported by Vertica. You can also specify the information of the attachment facility. Example: <code>jdbc:vertica://1*.0.0.1:3306/database</code> .</p>	No	None
username	The username for connecting to the Vertica database.	No	None
password	The password for connecting to the Vertica database.	No	None

Parameter	Description	Required	Default value
table	<p>The name of the source table from which Vertica Reader reads data. Vertica Reader can read data from multiple tables. The tables are described in a JSON array.</p> <p>If you specify multiple tables, make sure that the tables have the same schema. Vertica Reader does not check whether the tables have the same schema.</p> <div style="background-color: #e0f2f1; padding: 5px; border: 1px solid #ccc;"> <p> Note The table parameter must be included in the connection parameter.</p> </div>	Yes	None
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns in the source table.</p> <ul style="list-style-type: none"> • Column pruning is supported. You can select specific columns to export. • The column order can be changed. You can export the specified columns in an order different from that specified in the schema of the table. • Constants are supported. • The column parameter must explicitly specify a set of columns to be synchronized, and cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when Vertica Reader reads data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> • We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to specific shards. • The splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you set this parameter to a column of an unsupported type, Vertica Reader returns an error. • If you leave the splitPk parameter empty, Vertica Reader reads data from the source table by using a single thread. 	No	None
where	<p>The WHERE clause. Vertica Reader generates a SELECT statement based on the table, column, and where parameters that you have configured, and uses the generated SELECT statement to select and read data.</p> <p>For example, you can specify the where parameter during testing. To synchronize data generated on the current day, set the where parameter to <code>gmt_create > \$bizdate</code> .</p> <ul style="list-style-type: none"> • You can use the WHERE clause to synchronize incremental data. • If you do not specify the where parameter or leave it empty, all data is read. 	No	None

Parameter	Description	Required	Default value
querySql	<p>The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter.</p> <p>If you specify the querySql parameter, Vertica Reader ignores the table, column, and where parameters that you have configured.</p>	No	None
fetchSize	<p>The number of data records to read at a time. This parameter determines the number of interactions between Data Integration and the database and affects data reading efficiency.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note A value greater than 2048 may lead to OOM during the data synchronization process.</p> </div>	No	1024

Configure Vertica Reader by using the codeless UI

The codeless UI is not supported for Vertica Reader.

Configure Vertica Reader by using the code editor

In the following code, a node is configured to read data from a Vertica database.

```
{
  "type": "job",
  "steps": [
    {
      "stepType": "vertica", // The reader type.
      "parameter": {
        "datasource": "", // The connection name.
        "username": "",
        "password": "",
        "where": "",
        "column": [ // The columns to be synchronized.
          "id",
          "name"
        ],
        "splitPk": "id",
        "connection": [
          {
            "table": [ // The name of the table to be synchronized.
              "table"
            ],
            "jdbcUrl": [
              "jdbc:vertica://host:port/database"
            ]
          }
        ]
      },
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "stream",
      "parameter": {

```

```

    "parameter": {
      "print": false,
      "fieldDelimiter": ",",
    },
    "name": "Writer",
    "category": "writer"
  }
],
"version": "2.0",
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
},
"setting": {
  "errorLimit": {
    "record": "0" // The maximum number of dirty data records allowed.
  },
  "speed": {
    "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent": 1 // The maximum number of concurrent threads.
  }
}
}
}

```

2.4.5.3.22. Configure GBase Reader

This topic describes how GBase Reader reads data and how to configure a sync node to read data from a GBase database.

GBase Reader connects to a remote GBase database through the MySQL Java Database Connectivity (JDBC) Driver, generates SQL statements based on your configurations, and then reads data from the remote GBase database. Then, GBase Reader assembles the returned data to abstract datasets in custom data types supported by Data Integration, and passes the datasets to a writer.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized. You can select only one source table for each sync node.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be synchronized from the source table. The columns are described in a JSON array. The default value is [*], which indicates all columns.</p> <ul style="list-style-type: none"> • Column pruning is supported. You can select and export specific columns. • Change of the column order is supported. You can export the columns in an order different from that specified in the schema of the table. • Constants are supported. The column names must be arranged in compliance with the SQL syntax supported by MySQL. For example, <code>["id"."table"."1","mingya.wmy","null","to_char(a+1)","2.3","true"]</code>. <ul style="list-style-type: none"> ◦ id: a column name. ◦ table: the name of a column that contains reserved keywords. ◦ 1: an integer constant. ◦ 'mingya.wmy': a string constant, which is enclosed in single quotation marks (' '). ◦ null: <ul style="list-style-type: none"> ▪ ' ' indicates an empty value. ▪ null indicates a null value. ▪ 'null' indicates the string null. ◦ to_char(a+1): a function expression. ◦ 2.3: a floating-point constant. ◦ true: a Boolean value. • The column parameter must explicitly specify a set of columns to be synchronized. The parameter cannot be left empty. 	Yes	None
splitPk	<p>The field used for data sharding when GBase Reader extracts data. If you specify the splitPk parameter, the table is sharded based on the shard key specified by this parameter. Data Integration then runs concurrent threads to synchronize data. This improves efficiency.</p> <ul style="list-style-type: none"> • We recommend that you set the splitPk parameter to the primary key of the table. Based on the primary key, data can be well distributed to different shards, but not intensively distributed to certain shards. • Currently, the splitPk parameter supports data sharding only for integers but not for other data types such as string, floating point, and date. If you specify this parameter to a column of an unsupported type, GBase Reader ignores the splitPk parameter and synchronizes data through a single thread. • If you do not specify the splitPk parameter or leave it empty, Data Integration synchronizes data through a single thread. 	No	None

Parameter	Description	Required	Default value
where	<p>The WHERE clause. For example, set this parameter to <code>gmt_create>\$bizdate</code> .</p> <ul style="list-style-type: none"> You can use the WHERE clause to synchronize incremental data. If you do not specify the where parameter or leave it empty, all data is synchronized. Do not set the where parameter to limit 10, which does not conform to the constraints of MySQL on the SQL WHERE clause. 	No	None
querySql (only available in the code editor)	<p>The SELECT statement used for refined data filtering. If you specify this parameter, Data Integration directly filters data based on this parameter. For example, if you want to join multiple tables for data synchronization, set this parameter to <code>select a.b from table_a join table_b on table_a.id = table_b.id</code> . The priority of the querySql parameter is higher than those of the table, column, where, and splitPk parameters. If you specify the querySql parameter, GBase Reader ignores the table, column, where, and splitPk parameters that you have configured. The datasource parameter parses information, including the username and password, from this parameter.</p>	No	None

Configure GBase Reader by using the codeless UI

Currently, the codeless user interface (UI) is not supported for GBase Reader.

Configure GBase Reader by using the code editor

In the following code, a node is configured to read data from a GBase database.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"gbase // The reader type.
      "parameter":{
        "column":[// The columns to be synchronized.
          "id"
        ],
        "connection":[
          { "querysql":["select a,b from join1 c join join2 d on c.id = d.id;"], // Specify the querySql parameter in the connection parameter as a string.
            "datasource":"","// The connection name.
            "table":[// The name of the table to be synchronized.
              "xxx"
            ]
          }
        ],
        "where":"","// The WHERE clause.
        "splitPk":"","// The shard key.
        "encoding":"UTF-8">// The encoding format.
      },
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0">// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1,// The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

2.4.5.4. Configure the writer

2.4.5.4.1. Configure AnalyticDB for MySQL 2.0 Writer

This topic describes the data types and parameters supported by AnalyticDB for MySQL 2.0 Writer and how to configure it by using the codeless UI and code editor.

Prerequisites

Data Integration can import data to AnalyticDB for MySQL 2.0 in real time. This method requires you to create real-time tables, which are fact tables, in the destination AnalyticDB for MySQL 2.0 database in advance. In real-time import mode, data is imported efficiently and the process is simple.

You must configure a connection before you configure AnalyticDB for MySQL 2.0 Writer.

Data types

The following table lists the data types supported by AnalyticDB for MySQL 2.0 Writer.

Category	AnalyticDB for MySQL 2.0 data type
Integer	INT, TINYINT, SMALLINT, and BIGINT
Floating point	FLOAT and DOUBLE
String	VARCHAR
Date and time	DATE and TIME
Boolean	BOOLEAN

Parameters

Parameter	Description	Required	Default value
connectionUrl	The URL for connecting to the AnalyticDB for MySQL 2.0 database. Specify the parameter in the IP address:Port format.	Yes	None
database	The name of the AnalyticDB for MySQL 2.0 database.	Yes	None
Access Id	The AccessKey ID that you can use to connect to the AnalyticDB for MySQL 2.0 database.	Yes	None
Access Key	The AccessKey secret that you can use to connect to the AnalyticDB for MySQL 2.0 database.	Yes	None
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the destination table.	Yes	None
partition	The partition name of the destination table. If the destination table is partitioned, this parameter is required.	No	None
writeMode	The write mode. Set the value to insert. In this mode, if a primary key conflict occurs, the conflicting rows are overwritten.	Yes	None

Parameter	Description	Required	Default value
column	The columns in the destination table to which data is written. Separate the columns with commas (,), for example, ["a","b","c"]. Set the value to an asterisk (*) if data is written to all the columns in the destination table.	Yes	None
suffix	Optional. The suffix to the AnalyticDB for MySQL 2.0 URL that is in the format of IP address:Port . This suffix is a custom connection string. After this parameter is set, the URL changes to a JDBC connection string for connecting to AnalyticDB for MySQL 2.0. For example, set the suffix parameter to autoReconnect=true&failOverReadOnly=false&maxReconnects=10 .	No	None
batchSize	The number of data records to write at a time. This parameter is available only when the writeMode parameter is set to insert.	Required only when the writeMode parameter is set to insert	None
bufferSize	The size of the Data Integration data buffer, which is designed to improve the performance of AnalyticDB for MySQL 2.0. Data from the source database is sorted in the buffer before the data is committed to AnalyticDB for MySQL 2.0. The data in the buffer is sorted based on the partition key columns in AnalyticDB for MySQL 2.0. In this way, the data is organized in an order that can improve the performance of the AnalyticDB for MySQL 2.0 server. Data in the buffer is committed to AnalyticDB for MySQL 2.0 in batches based on the batchSize parameter. We recommend that you set the bufferSize value to a multiple of batchSize. This parameter is available only when the writeMode parameter is set to insert.	Required only when the writeMode parameter is set to insert	Disabled

Configure AnalyticDB for MySQL 2.0 Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Write Method	The writeMode parameter in the preceding parameter description.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.

Configure AnalyticDB for MySQL 2.0 Writer by using the code editor

```
{
  "type": "job",
  "version": "2.0",
  "steps": [
    {
      "stepType": "stream",
      "parameter": {
        "name": "Reader",
        "category": "reader"
      }
    },
    {
      "stepType": "ads", // The writer type.
      "parameter": {
        "partition": "", // The partition name of the destination table.
        "datasource": "", // The connection name.
        "column": [ // The columns to which data is written.
          "id"
        ],
        "writeMode": "insert", // The write mode.
        "batchSize": "256", // The number of data records to write at a time.
        "table": "", // The name of the destination table.
        "overwrite": "true" // Specifies whether to overwrite the destination table when data is written to AnalyticDB for MySQL 2.0. A value of true indicates that the destination table is overwritten. A value of false indicates that the destination table is not overwritten and the new data is appended to the existing data. This value takes effect only when the writeMode parameter is set to load.
      }
    },
    {
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent": 1 // The maximum number of concurrent threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}
```

2.4.5.4.2. Configure DataHub Writer

This topic describes the data types and parameters supported by DataHub Writer and how to configure it by using the code editor.

DataHub is a real-time data distribution platform designed to process streaming data. You can publish and subscribe applications to streaming data in DataHub and distribute the data to other platforms. This allows you to easily analyze streaming data and build applications based on the streaming data.

Based on the Apsara system of Alibaba Cloud, DataHub features high availability, low latency, high scalability, and high throughput. Seamlessly integrated with Realtime Compute, DataHub allows you to easily use SQL to analyze streaming data. DataHub can also distribute streaming data to Alibaba Cloud services such as MaxCompute and OSS.

 **Note** Strings can only be UTF-8 encoded. The size of each string must not exceed 1 MB.

Parameter configuration

The source is connected to the destination through a single channel. Therefore, the channel type configured for the writer must be the same as that configured for the reader. Generally, channels are categorized into two types: memory and file. The following configuration sets the channel type to file:

```
"agent.sinks.dataXSinkWrapper.channel": "file"
```

Parameters

Parameter	Description	Required	Default value
accessId	The AccessKey ID for accessing DataHub.	Yes	None
accessKey	The AccessKey secret for accessing DataHub.	Yes	None
endpoint	The endpoint of DataHub.	Yes	None
maxRetryCount	The maximum number of retries if a task fails.	No	None
mode	The mode for writing strings.	Yes	None
parseContent	The data that has been parsed.	Yes	None
project	The organizational unit in DataHub. Each project contains one or more topics.  Note DataHub projects are independent from MaxCompute projects. Projects created in MaxCompute cannot be used in DataHub.	Yes	None
topic	The minimum unit for data subscription and publication. You can use topics to distinguish different types of streaming data.	Yes	None
maxCommitSize	The amount of data, in MB, that DataHub Writer buffers before sending it to the destination. This mechanism aims to improve writing efficiency. The default value is 1048576, in KB, that is, 1 MB.	No	1048576

Parameter	Description	Required	Default value
batchSize	The number of data records that DataHub Writer buffers before sending them to the destination. This mechanism aims to improve writing efficiency. The default value is 1024.	No	1,024
maxCommitInterval	The maximum interval at which DataHub Writer sends data to the destination. When an interval ends, DataHub Writer sends buffered data even if the data amount does not reach the preceding two thresholds. The default value is 30000, in milliseconds, that is, 30 seconds.	No	30,000
parseMode	The mode for parsing log entries. Valid values: <i>default</i> and <i>csv</i> . The value <i>default</i> indicates that no log parsing is required. The value <i>csv</i> indicates that a delimiter is inserted between fields for each log entry.	No	<i>default</i>

Configure DataHub Writer by using the codeless UI

Currently, the codeless UI is not supported for DataHub Writer.

Configure DataHub Writer by using the code editor

In the following code, a node is configured to read data from the memory and then write the data to DataHub.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "datahub", // The writer type.
      "parameter": {
        "datasource": "", // The connection name.
        "topic": "" // The minimum unit for data subscription and publication. You can use topics to distinguish different types of streaming data.
      },
      "maxRetryCount": 500, // The maximum number of retries if a task fails.
      "maxCommitSize": 1048576 // The amount of data, in MB, that DataHub Writer buffers before sending it to the destination.
      "shardId": "xxxxxx" // The shard of the DataHub topic.
    },
    {
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "concurrent": 20, // The maximum number of concurrent threads.
      "throttle": false, // The value false indicates that the bandwidth is not throttled. The value true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.4.3. Configure the DB2 writer

The DB2 writer enables writing data to tables stored on Db2 databases. To write data into a Db2 table, the DB2 writer connects to the remote Db2 database through JDBC, and runs `INSERT INTO` statements. Data is written into the Db2 table in batches.

The DB2 writer is designed for ETL developers to import data from data warehouses to Db2 databases. It also serves as a data migration tool for database administrators and other users.

The DB2 writer reads data from the channel, connects to a remote Db2 database through JDBC, and then runs `INSERT INTO` statements. The rows that violate the unique index constraint or primary key constraint cannot be written into the Db2 database. To improve performance, the DB2 writer makes batch updates with the `PreparedStatement` method and sets `rewriteBatchedStatements=true`. In this way, the DB2 writer buffers data, and submits a write request when the amount of data in the buffer reaches a specific threshold.

Note The `INSERT INTO` privilege is required for data synchronization tasks with the DB2 writer. Whether other permissions are required depends on the SQL statements specified in the `preSql` and `postSql` parameters.

The DB2 writer supports most Db2 data types. Since still some of the Db2 data types are not supported, verify that your data types are supported.

The following table lists data types supported by the DB2 writer.

Data Integration data type	Db2 data type
Integer	SMALLINT
Floating point	DECIMAL, REAL, and DOUBLE
String	CHAR, CHARACTER, VARCHAR, GRAPHIC, VARGRAPHIC, LONG VARCHAR, CLOB, LONG VARGRAPHIC, and DBCLOB
Date and time	DATE, TIME, and TIMESTAMP
Boolean	N/A
Binary	BLOB

Parameters

Parameter	Description	Required	Default value
<code>jdbcUrl</code>	The JDBC connectivity URL, used to connect to the Db2 database. In accordance with Db2 official specifications, the URL format must be <code>jdbc:db2://ip:port/database</code> . You can also specify the information of the attachment facility.	Yes	None
<code>username</code>	The username used to connect to the data source.	Yes	None
<code>password</code>	The password used to connect to the data source.	Yes	None
<code>table</code>	The name of the destination table.	Yes	None
<code>column</code>	The columns in the destination table to which data is written. Separate the columns with a comma (,). Example: <code>"column": ["id", "name", "age"]</code> . Set the value to an asterisk (*) if data is written to all the columns in the destination table. Example: <code>"column": ["*"]</code> .	Yes	None
<code>preSql</code>	The SQL statement runs before the data synchronization task starts. Currently, you can run only one SQL statement. For example, you can run a statement to clear outdated data.	No	None

Parameter	Description	Required	Default value
postSql	The SQL statement runs after the data synchronization task ends. Currently, you can run only one SQL statement in wizard mode but multiple SQL statements in script mode. For example, you can run a statement to add a timestamp.	No	None
batchSize	The number of data records to write per batch. Setting this parameter can greatly reduce the interactions between Data Integration and the Db2 database over the network, and increase the throughput. However, an excessively large value may cause the running Data Integration process to become out of memory (OOM).	No	1024

Configure the DB2 writer in wizard mode

Currently, wizard mode is not supported for the DB2 writer.

Configure the DB2 writer in script mode

In the following script, a task is configured to write data to a Db2 database.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    // The following template is used to configure the reader. For more information, see the corresponding section
    .
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "db2", // The writer type.
      "parameter": {
        "postSql": [], // The SQL statement runs after the data synchronization task ends.
        "password": "", // The password.
        "jdbcUrl": "jdbc:db2://ip:port/database", //The JDBC connectivity URL, used to connect to the Db2 database.
        "column": [
          "id"
        ],
        "batchSize": 1024, // The number of data records to write per batch.
        "table": "", // The table name.
        "username": "", // The username.
        "preSql": [] // The SQL statement runs before the data synchronization task starts.
      },
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // The value false means that the bandwidth is not throttled. The value true means that the b
      andwidth is throttled. The maximum transmission rate takes effect only if you specify this parameter as true.
      "concurrent": 1, // The maximum number of concurrent threads.
      "dmu": 1 // The number of DMUs.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}
```

2.4.5.4.4. Configure DRDS Writer

This topic describes the data types and parameters supported by DRDS Writer and how to configure it by using the codeless UI and code editor.

DRDS Writer allows you to write data to tables stored in DRDS databases. DRDS Writer connects to the proxy of a remote DRDS database by using JDBC, and executes a `REPLACE INTO` statement to write data to the DRDS database.

 **Note**

- To execute the `REPLACE INTO` statement, make sure that your table has the primary key or a unique index to avoid replicated data.
- You must configure a connection before you configure DRDS Writer.

DRDS Writer is designed for ETL developers to import data from data warehouses to DRDS databases. DRDS Writer can also be used as a data migration tool by users such as DBAs.

DRDS Writer obtains data from a Data Integration reader, and writes the data to the destination database by executing the `REPLACE INTO` statement. If no primary key conflict or unique index conflict occurs, the action is the same as that of the `INSERT INTO` statement. If a conflict occurs, original rows are replaced by new rows. DRDS Writer sends data to the DRDS proxy when the amount of buffered data reaches a specific threshold. The proxy determines whether to write the data to one or more tables and how to route the data when it is written to multiple tables.



Note A sync node that uses DRDS Writer must have at least the permission to execute the `REPLACE INTO` statement. Whether other permissions are required depends on the SQL statements specified in the `preSql` and `postSql` parameters.

Similar to MySQL Writer, DRDS Writer supports most MySQL data types. Make sure that your data types are supported.

The following table lists the data types supported by DRDS Writer.

Category	DRDS data type
Integer	INT, TINYINT, SMALLINT, MEDIUMINT, BIGINT, and YEAR
Floating point	FLOAT, DOUBLE, and DECIMAL
String	VARCHAR, CHAR, TINYTEXT, TEXT, MEDIUMTEXT, and LONGTEXT
Date and time	DATE, DATETIME, TIMESTAMP, and TIME
Boolean	BIT and BOOLEAN
Binary	TINYBLOB, MEDIUMBLOB, BLOB, LONGBLOB, and VARBINARY

Parameters

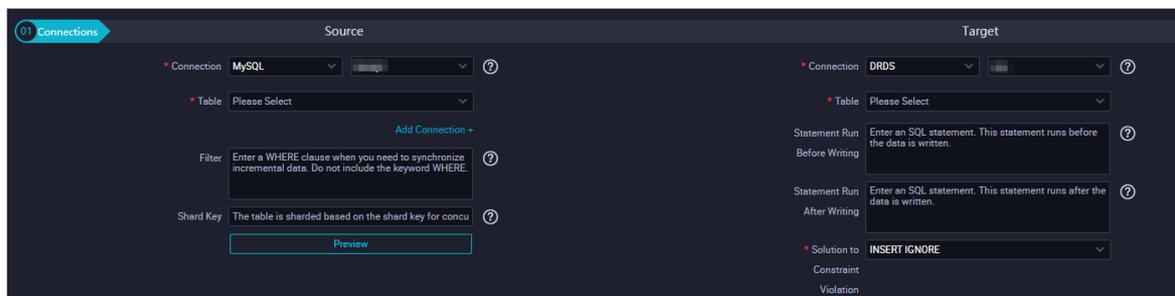
Parameter	Description	Required	Default value
<code>datasource</code>	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
<code>table</code>	The name of the destination table.	Yes	None

Parameter	Description	Required	Default value
writeMode	<p>The write mode. Valid values: <i>insert into</i>, <i>on duplicate key update</i>, and <i>replace into</i>.</p> <ul style="list-style-type: none"> <i>insert into</i>: If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows and is regarded as dirty data. <i>on duplicate key update</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of <i>insert into</i>. If a conflict occurs, specified fields in original rows are updated. <i>replace into</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of <i>insert into</i>. If a conflict occurs, original rows are deleted and new rows are inserted. This means that all the field values of the original rows are replaced. 	No	<i>insert</i>
column	The columns in the destination table to which data is written. Separate the columns with commas (,), for example, "column":["id","name","age"]. To write data to all the columns in the destination table, set the value to an asterisk (*), for example, "column":["*"].	Yes	None
preSql	The SQL statement to execute before the sync node is run. For example, you can clear outdated data before data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
postSql	The SQL statement to execute after the sync node is run. For example, you can add a timestamp after data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the DRDS database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,024

Configure DRDS Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.



Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.

Parameter	Description
Table	The table parameter in the preceding parameter description.
Statement Run Before Writing	The preSql parameter in the preceding parameter description. Enter an SQL statement to execute before the sync node is run.
Statement Run After Writing	The postSql parameter in the preceding parameter description. Enter an SQL statement to execute after the sync node is run.
Solution to Constraint Violation	The writeMode parameter in the preceding parameter description. Select the required write mode.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.

Configure DRDS Writer by using the code editor

In the following code, a node is configured to write data to a DRDS database.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"drds",// The writer type.
      "parameter":{
        "postSql":[// The SQL statement to execute after the sync node is run.
          "datasource":"","// The connection name.
          "column":[// The columns to which data is written.
            "id"
          ],
          "writeMode":"insert ignore",
          "batchSize":"1024",// The number of data records to write at a time.
          "table":"test",// The name of the destination table.
          "preSql":[// The SQL statement to execute before the sync node is run.
        ],
        "name":"Writer",
        "category":"writer"
      }
    },
    "setting":{
      "errorLimit":{
        "record":"0",// The maximum number of dirty data records allowed.
      },
      "speed":{
        "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
        "concurrent":1 // The maximum number of concurrent threads.
      }
    },
    "order":{
      "hops":[
        {
          "from":"Reader",
          "to":"Writer"
        }
      ]
    }
  }
}
```

2.4.5.4.5. Configure the FTP writer

The FTP writer allows you to write one or more files in CSV format into a remote FTP file. At the underlying level, this writer converts the data that is readable by the Data Integration service to CSV files, and writes these files into the remote FTP server using FTP network protocols. You must configure the data source before configuring the FTP writer.

 **Note** For more information, see [Add FTP data sources](#).

The FTP writer can only write data into FTP files that store logical two-dimensional tables, for example, text information in the CSV format.

This writer enables you to convert data that is readable by the Data Integration service to FTP files. FTP files stores non-structured data. The advantages and disadvantages of the FTP writer are described as follows:

- Only supports text files and the schema in the text file must be a two-dimensional table. It does not support the blob type, such as video data.
- Supports CSV and text files with custom delimiters.
- Does not support text compression when data is written to the destination table.
- Supports multi-thread writing, with each thread performing write operations on a subfile.

Currently, the FTP writer does not support the following two features:

- Concurrent writing for a single file.
- Providing varying data types. The FTP does not provide data types, and the FTP writer writes data of the string type into FTP files.

Parameters

Parameter	Description	Required.	Default value
datasource	The name of the data source. You can add a data source either in wizard or script mode. The value of this parameter must be identical to the real data source name.	Yes	None
timeout	The timeout period for the connection to the FTP server, measured in milliseconds.	No	60000 (1 minute)
path	The path of the FTP file system. The write can write data into multiple files in the path.	Yes	None
fileName	The name of the file into which data is written. A random suffix is added to the file name to form the actual name of the file into which the data is written on each thread.	Yes	None
writeMode	The mode in which the FTP writer clears existing data before writing data. Valid values: <ul style="list-style-type: none"> • truncate: The writer clears all the files prefixed by fileName in the path before writing data. • append: No processing is performed on the file before the FTP writer imports data into this file. In the Data Integration service, the FTP writer uses the original file name in the data source. No duplicate file names are allowed. • nonConflict: An error is reported if a file prefixed by fileName exists in the path. 	Yes	None
fieldDelimiter	The column delimiter of the file to be written.	Yes. A single character is used.	None

Parameter	Description	Required.	Default value
compress	The compress option. The gzip and bzip2 compression options are supported.	No	No
encoding	The encoding of the file to be read.	No	UTF-8
nullFormat	The string that represents null. Since no standard strings can represent null in text files, Data Integration provides the nullFormat parameter to define which string represents a null pointer. For example, if you specify <code>nullFormat:"null"</code> , Data Integration considers "null" as a null pointer.	No	None
dateFormat	The date format, for example, "dateFormat": "yyyy-MM-dd".	No	None
fileFormat	The file format, including CSV and text. For the CSV format, if you want to write the data that includes column delimiters, the delimiters are escaped with quotation marks. For text format, the data to be written is separated by column delimiters without being escaped.	No	text
header	The header used when a txt file is written, for example, ['id', 'name', 'age'].	No	None
Markdonefilename	The name of the file marked as "done". After a synchronization task is completed, a MarkDoneFile is generated, based on which you can determine whether the task is executed successfully.	No	None

Configure the FTP writer in wizard mode

1. Select data sources.

Configure the source and destination for the data synchronization task.

Parameter	Description
Data Source	The datasource parameter provided in the preceding table. Select a data source type, and enter the name of a data source that has been configured in DataWorks.
File Path	The path parameter provided in the preceding table.
Column Delimiter	The fieldDelimiter parameter provided in the preceding table. Default value: a comma (,)
Encoding	The encoding parameter provided in the preceding table. Default value: UTF-8.
Null String	The nullFormat parameter provided in the preceding table, which defines a string that represents null.
Compression Format	The nullFormat parameter provided in the preceding table. Default value: No.
Include Header	The skipHeader parameter in the preceding table. Default value: No.

Parameter	Description
Prefix Conflict	The writeMode parameter provided in the preceding table, which defines a string that represents null.

2. Configure field mappings. It is equivalent to setting the column parameter provided in the preceding table.

You can map the left-side source table fields to the right-side destination table fields. You can also click **Add** to add a field or click the **Delete** icon to delete a field in the source table.

After you click **Map Fields in the Same Line**, each source table field is mapped to the destination table field in the same line if exists. Ensure that the conversion between data types is feasible.

3. Configure the channel.

Parameter	Description
DMU	The data processing capabilities. A data migration unit (DMU) represents the data processing capabilities for data integration, given a minimum configuration of the CPU, memory, network, and other resources.
Concurrent Jobs	The maximum number of concurrent threads to read and write data to data storage within the data synchronization task.
Transmission Rate	You can throttle the bandwidth and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Task Resource Group	The servers on which tasks are run. If an excessively large number of tasks are run on the default resource group, some tasks may be delayed due to insufficient resources. In this case, you can configure additional servers.

Configure the FTP writer in script mode

In the following script, a task is configured to write data to an FTP database.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    { // The following template is used to configure the reader. For more information, see the corresponding section
      "stepType":"stream",
      "Parameter":{
        "name":"Reader",
        "category":"reader"
      },
      {
        "stepType":"ftp",// The plug-in name.
        "parameter":{
          "path":"","// The file path.
          "fileName":"","// The file name.
          "nullFormat": "null", // The string that represents null.
          "dateFormat":"yyyy-MM-dd HH:mm:ss", // The time format.
          "datasource":""," // The data source.
          "writeMode":"","// The writing method.
          "fieldDelimiter":",", // The column delimiter.
          "encoding":"","// The encoding.
          "fileFormat":"","// The file type.
        },
        "name":"Writer",
        "category":"writer"
      }
    ],
    "setting":{
      "errorLimit":{
        "record":"0",// The maximum number of dirty data records allowed.
      },
      "speed":{
        "throttle":false, // The value false means that the bandwidth is not throttled. The value true means that the bandwidth is throttled. The maximum transmission rate takes effect only if you specify this parameter as true.
        "concurrent":"1", // The maximum number of concurrent threads.
        "dmu":1 // The number of DMUs.
      }
    },
    "order":{
      "hops":[
        {
          "from":"Reader",
          "to":"Writer"
        }
      ]
    }
  }
}
```

2.4.5.4.6. Configure HBase Writer

This topic describes the features, data types, and parameters supported by HBase Writer and how to configure it by using the code editor.

HBase Writer allows you to write data to HBase data stores. Specifically, HBase Writer connects to a remote HBase data store through the Java client of HBase. Then, HBase Writer uses the PUT method to write data to the HBase data store.

Features

- HBase 0.94.x and 1.1.x are supported.
 - If you use HBase 0.94.x, set the hbaseVersion parameter to 094x for the writer.

```
"writer": {
  "hbaseVersion": "094x"
}
```

- If you use HBase 1.1.x, set the hbaseVersion parameter to 11x for the writer.

```
"writer": {
  "hbaseVersion": "11x"
}
```

 **Note** Currently, HBase Writer for HBase 1.1.x is compatible with HBase 2.0. If you have any issues in using HBase Writer with HBase 2.0, submit a ticket.

- You can use concatenated fields as a rowkey.

Currently, HBase Writer supports concatenating multiple fields to generate the rowkey of an HBase table.
- You can set the version of each HBase cell.

The information that can be used as the version of an HBase cell includes:

 - Current time
 - Specified source column
 - Specified time

Data types

The following table lists the data types supported by HBase Writer.

-  **Note**
- The types of the specified columns must be the same as those in the HBase table.
 - Data types that are not listed in the table are not supported.

Category	HBase data type
Integer	Int, Long, and Short
Floating point	Float and Double
Boolean	Boolean
String	String

Parameters

Parameter	Description	Required	Default value
haveKerberos	<p>Specifies whether Kerberos authentication is required. A value of true indicates that Kerberos authentication is required.</p> <div style="background-color: #e1f5fe; padding: 10px; border: 1px solid #cfe2f3;"> <p>Note</p> <ul style="list-style-type: none"> • If the value is true, the following five Kerberos-related parameters must be specified: <ul style="list-style-type: none"> ◦ kerberosKeytabFilePath ◦ kerberosPrincipal ◦ hbaseMasterKerberosPrincipal ◦ hbaseRegionserverKerberosPrincipal ◦ hbaseRpcProtection • If the value is false, Kerberos authentication is not required and you do not need to specify the preceding parameters. </div>	No	false
hbaseConfig	The properties of the HBase cluster, in JSON format. The hbase.zookeeper.quorum parameter is required. It specifies the ZooKeeper ensemble servers. You can also configure other properties, such as those related to the cache and batch for scan operations.	Yes	None
mode	The mode in which data is written to the HBase data store. Currently, only the normal mode is supported. The dynamic column selection mode is coming soon.	Yes	None
table	The name of the HBase table to which data is written. The name is case-sensitive.	Yes	None
encoding	The encoding format in which a string is converted through byte[]. Currently, UTF-8 and GBK are supported.	No	utf-8
column	<p>The HBase columns to which data is written.</p> <ul style="list-style-type: none"> • index: the ID of the column in the source table, starting from 0. • name: the name of the column in the HBase table, in the columnFamily:column format. • type: the type of the data written, which is used by the byte[] constructor. 	Yes	None
maxVersion	The number of versions read by HBase Reader when multiple versions are available. Valid values: -1 and integers greater than 1. A value of -1 indicates that all versions are read.	Required in multiVersionFixedColumn mode	None

Parameter	Description	Required	Default value
range	<p>The rowkey range that HBase Reader reads.</p> <ul style="list-style-type: none"> • startRowkey: the start rowkey. • endRowkey: the end rowkey. • isBinaryRowkey: the operation called by byte[] to convert the specified start and end rowkeys. Default value: false. If the value is true, Bytes.toBytesBinary(rowkey) is called. If the value is false, Bytes.toBytes(rowkey) is called. Example: <pre data-bbox="416 568 1114 752">"range": { "startRowkey": "aaa", "endRowkey": "ccc", "isBinaryRowkey": false }</pre> <p>Example:</p> <pre data-bbox="416 808 1114 1216">"column": [{ "index": 1, "name": "cf1:q1", "type": "string" }, { "index": 2, "name": "cf1:q2", "type": "string" }]</pre>	No	None

Parameter	Description	Required	Default value
rowkeyColumn	<p>The rowkey of each HBase cell.</p> <ul style="list-style-type: none"> index: the ID of the column in the source table, starting from 0. If the column is a constant, set the value to -1. type: the type of the data written, which is used by the byte[] constructor. value: a constant, which is usually used as the delimiter between fields. HBase Writer sequentially concatenates all columns specified in this parameter to a string, and uses the string as the rowkey. The specified columns cannot be all constants. <p>Example:</p> <pre> "rowkeyColumn": [{ "index":0, "type":"string" }, { "index":-1, "type":"string", "value":"_" }] </pre>	Yes	None
versionColumn	<p>The version of each HBase cell. You can use the current time, a specified source column, or a specified time as the version. If you do not specify this parameter, the current time is used.</p> <ul style="list-style-type: none"> index: the ID of the column in the source table, starting from 0. Make sure that the value can be properly converted to the Long type. type: the data type. If the type is Date, HBase Writer converts the date to yyyy-MM-dd HH:mm:ss or yyyy-MM-dd HH:mm:ss SSS. If you want to use a specified time as the version, set the value to -1. value: the specified time of the Long type. <p>Example:</p> <pre> "versionColumn":{ "index":1 } "versionColumn":{ "index":-1, "value":123456789 } </pre>	No	None

Parameter	Description	Required	Default value
nullMode	The method of processing null values. Valid values: <ul style="list-style-type: none"> skip: HBase Writer does not write null values to the HBase data store. empty: HBase Writer writes HConstants.EMPTY_BYTE_ARRAY (new byte [0]) to the HBase data store instead of null values. 	No	skip
walFlag	Specifies whether to enable write ahead logging (WAL) for HBase. If the value is true, all edits requested by an HBase client for all Regions carried by the RegionServer are recorded first in the WAL (that is, the HLog). After the edits are successfully recorded in the WAL, they are implemented to the Memstore and a success indication is sent to the HBase client. If edits fail to be recorded in the WAL, a failure indication is sent to the HBase client without implementing the edits. If the value is false, WAL is disabled but writing efficiency is improved.	No	<i>false</i>
writeBufferSize	The write buffer size, in bytes, of the HBase client. If you specify this parameter, you must also specify the autoflush parameter. autoflush: <ul style="list-style-type: none"> If the value is true, the HBase client sends a PUT request each time it receives an edit. If the value is false, the HBase client sends a PUT request only when its write buffer is full. 	No	8 MB

Configure HBase Writer by using the codeless UI

Currently, the codeless user interface (UI) is not supported for HBase Writer.

Configure HBase Writer by using the code editor

In the following code, a node is configured to write data to an HBase 1.1.x data store.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "hbase", // The writer type.
      "parameter": {
        "mode": "normal", // The mode in which data is written to the HBase data store.
        "walFlag": "false", // WAL is disabled for HBase.
        "hbaseVersion": "094x", // The HBase version.
        "rowkeyColumn": [ // The rowkey of each HBase cell.
          {
            "index": "0", // The ID of the column in the source table.
            "type": "string" // The data type.
          }
        ],
      }
    }
  ]
}
```

```
{
  "index": "-1",
  "type": "string",
  "value": "_ "
}
],
>nullMode: "skip", // The method of processing null values.
>column: [ // The HBase columns to which data is written.
  {
    "name": "columnFamilyName1:columnName1", // The name of the HBase column.
    "index": "0", // The ID of the column in the source table.
    "type": "string" // The data type.
  },
  {
    "name": "columnFamilyName2:columnName2",
    "index": "1",
    "type": "string"
  },
  {
    "name": "columnFamilyName3:columnName3",
    "index": "2",
    "type": "string"
  }
],
>writeMode: "api", // The write mode.
>encoding: "utf-8", // The encoding format.
>table: "", // The name of the destination table.
>hbaseConfig: { // The properties of the HBase cluster, in JSON format.
  "hbase.zookeeper.quorum": "hostname",
  "hbase.rootdir": "hdfs://ip:port/database",
  "hbase.cluster.distributed": "true"
}
},
>name: "Writer",
>category: "writer"
}
],
>setting: {
  "errorLimit": {
    "record": "0" // The maximum number of dirty data records allowed.
  },
  "speed": {
    "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent": "1" // The maximum number of concurrent threads.
  }
},
>order: {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}
```

2.4.5.4.7. Configure HBase11xsql Writer

This topic describes the features, data types, and parameters supported by HBase11xsql Writer and how to configure it by using the code editor.

Background information

HBase11xsql Writer allows you to write data in batches to HBase tables created by using Phoenix. Phoenix can encode the primary key to rowkey. If you directly use the HBase API to write data to an HBase table that is created by using Phoenix, you must manually convert data, which is troublesome and error-prone. HBase11xsql Writer allows you to write data to HBase tables that packs all values into a single cell per column family.

HBase11xsql Writer connects to a remote HBase data store by using JDBC, and executes an UPSERT statement to write data to the HBase data store.

Limits

- The column order specified in the writer must match that specified in the reader. When you configure the column order in the reader, you specify the order of columns in each row for the output data. When you configure the column order in the writer, you specify the expected order of columns for the input data.

Example:

Column order specified in the reader: c1, c2, c3, c4.

Column order specified in the writer: x1, x2, x3, x4.

In this case, the value of column c1 is assigned to column x1 in the writer. If the column order specified in the writer is x1, x2, x4, x3, the value of column c3 is assigned to column x4 and the value of column c4 is assigned to column x3.

- HBase11xsql Writer can write data only to HBase 1.x.
- HBase11xsql Writer can write data only to tables created by using Phoenix but not native HBase tables.
- HBase11xsql Writer cannot write data with timestamps.

Features

HBase11xsql Writer can write data of an indexed table and synchronously update all indexed tables.

How it works

HBase11xsql Writer connects to an HBase data store by using Phoenix, which is a JDBC driver, and executes an UPSERT statement to write data in batches to the destination table. Phoenix allows to synchronously update indexed tables when you write data.

Parameters

Parameter	Description	Required	Default value
plugin	The writer type. Set this value to hbase11xsql.	Yes	None
table	The name of the destination table. The name is case-sensitive. Generally, the name of a table that is created by using Phoenix consists of uppercase letters.	Yes	None

Parameter	Description	Required	Default value
column	<p>The name of the column. The name is case-sensitive. Generally, the name of each column in a table that is created by using Phoenix consists of uppercase letters.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> • HBase11xsql Writer writes data strictly in accordance with the order of the columns obtained from the reader. • You do not need to specify the data type for each column. HBase11xsql Writer automatically obtains the metadata of columns from Phoenix. </div>	Yes	None
hbaseConfig	<p>The properties of the HBase cluster. The <code>hbase.zookeeper.quorum</code> parameter is required. It specifies the ZooKeeper ensemble servers.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> • Separate the IP addresses with commas (,), for example, <code>ip1,ip2,ip3</code>. • The <code>zookeeper.znode.parent</code> parameter is optional. Default value: <code>/hbase</code>. </div>	Yes	None
batchSize	The number of data records to write at a time.	No	256
nullMode	<p>The method of processing null values. Valid values:</p> <ul style="list-style-type: none"> • <i>skip</i>: HBase11xsql Writer does not write null values to the HBase data store. • <i>empty</i>: HBase11xsql Writer writes 0 or an empty string instead of null values to the HBase data store. For a column of the numeric type, HBase11xsql Writer writes 0. For a column of the VARCHAR type, HBase11xsql Writer writes an empty string. 	No	<i>skip</i>

Configure HBase11xsql Writer by using the code editor

In the following code, a node is configured to write data to an HBase database.

```

{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "setting": {
      "errorLimit": {
        "record": "0"
      },
      "speed": {
        "mbps": "1",
        "concurrent": "1"
      }
    },
    "reader": {
      "plugin": "odps",
      "parameter": {
        "datasource": "",
        "table": "",
        "column": [],
        "partition": ""
      }
    },
    "plugin": "hbase11xsq",
    "parameter": {
      "table": "The case-sensitive name of the destination table",
      "hbaseConfig": {
        "hbase.zookeeper.quorum": "The IP addresses of ZooKeeper ensemble servers of the destination HBase cluster. Obtain the IP addresses from product engineers (PEs).",
        "zookeeper.znode.parent": "The root znode of the destination HBase cluster. Obtain the IP addresses from PEs."
      },
      "column": [
        "columnName"
      ],
      "batchSize": 256,
      "nullMode": "skip"
    }
  }
}

```

FAQ

Q: What is the proper number of concurrent threads? Can I increase the number of concurrent threads to speed up the synchronization?

A: In the data import process, the default size of a JVM heap is 2 GB. Concurrent synchronization requires multiple threads. However, excessive threads sometimes cannot speed up the synchronization and may even deteriorate the performance because of frequent garbage collection (GC). We recommend that you set the number of concurrent threads within the range from 5 to 10.

Q: What is the proper value for the batchSize parameter?

A: The default value of the batchSize parameter is 256. You can set a proper value for the batchSize parameter based on the data volume of each row. Generally, the data volume of each write operation is 2 MB to 4 MB. You can set the value to the data volume of a write operation divided by the data volume of a row.

2.4.5.4.8. Configure HDFS Writer

This topic describes the data types and parameters supported by HDFS Writer and how to configure it by using the code editor.

HDFS Writer allows you to write text, ORC, or Parquet files to the specified directory in HDFS. In addition, you can associate the fields in the files with those in Hive tables. You must configure a connection before you configure HDFS Writer.

How it works

HDFS Writer writes files to HDFS in the following way:

1. Creates a temporary directory that does not exist in HDFS based on the path parameter you specified.
The name of the temporary directory is in the format of path_Random suffix.
2. Writes files that are read by a Data Integration reader to the temporary directory.
3. Moves the files from the temporary directory to the specified directory in HDFS after all the files are written. HDFS Writer ensures that the file names do not conflict with existing files in HDFS when it moves the files.
4. Deletes the temporary directory. If the deletion is interrupted because HDFS Writer fails to connect to HDFS, you must manually delete the temporary directory.

 **Note** To synchronize data, use an administrator account with the read and write permissions.

Limits

- HDFS Writer can write only text, ORC, and Parquet files that store logical two-dimensional tables to HDFS.
- HDFS is a distributed file system and does not have a schema. Therefore, you cannot write only some of the columns in a file to HDFS.
- HDFS Writer supports only the following Hive data types:
 - Numeric: TINYINT, SMALLINT, INT, BIGINT, FLOAT, and DOUBLE
 - String: STRING, VARCHAR, and CHAR
 - Boolean: BOOLEAN
 - Date and time: DATE and TIMESTAMP
- HDFS Writer does not support other Hive data types, such as DECIMAL, BINARY, ARRAY, MAP, STRUCT, or UNION.
- HDFS Writer can write data to only one partition in a partitioned Hive table at a time.
- To write a text file to HDFS, make sure that the delimiter in the file is the same as that in the Hive table to be associated with the file. Otherwise, you cannot associate the fields in the file stored in HDFS with those in the Hive table.
- HDFS Writer can be used in the environment where Hive 1.1.1 and Hadoop 2.7.1 (JDK version: 1.7) are installed. HDFS Writer can write files to HDFS properly in testing environments where Hadoop 2.5.0, Hadoop 2.6.0, or Hive 1.2.0 is installed.

Data types

HDFS Writer supports most Hive data types. Make sure that your data types are supported.

The following table lists the Hive data types supported by HDFS Writer.

 **Note** The types of the specified columns must be the same as those of columns in the Hive table.

Category	Hive data type
Integer	TINYINT, SMALLINT, INT, and BIGINT
Floating point	FLOAT and DOUBLE
String	CHAR, VARCHAR, and STRING
Boolean	BOOLEAN
Date and time	DATE and TIMESTAMP

Parameters

Parameter	Description	Required	Default value
defaultFS	The address of the HDFS NameNode, for example, <code>hdfs://127.0.0.1:9000</code> . The default resource group does not support configuring advanced Hadoop parameters related to the high availability feature.	Yes	None
fileType	The format of the files to be written to HDFS. Valid values: <ul style="list-style-type: none"> <code>text</code>: the text file format. <code>orc</code>: the ORC file format. <code>parquet</code>: the common Parquet file format. 	Yes	None
path	The directory in HDFS to which the files are written. HDFS Writer concurrently writes multiple files to the directory based on the concurrency setting. To associate the fields in a file with those in a Hive table, set the path parameter to the storage path of the Hive table in HDFS. Assume that the storage path specified for the data warehouse of Hive is <code>/user/hive/warehouse/</code> . The storage path of the hello table created in the test database is <code>/user/hive/warehouse/test.db/hello</code> .	Yes	None
fileName	The name prefix of the files to be written to HDFS. A random suffix is appended to the specified prefix to form the actual file name used by each thread.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns to be written to HDFS. You cannot write only some of the columns in a file to HDFS.</p> <p>To associate the fields in a file with those in a Hive table, specify the name and type parameters for each field.</p> <p>You can also specify the column parameter in the following way:</p> <pre>"column": [{ "name": "userName", "type": "string" }, { "name": "age", "type": "long" }]</pre>	Yes (Not required if the fileType parameter is set to parquet)	None
writeMode	<p>The mode in which HDFS Writer writes the files. Valid values:</p> <ul style="list-style-type: none"> <i>append</i>: writes the files based on the specified file name prefix and ensures that the actual file names do not conflict with those of existing files. <i>nonConflict</i>: returns an error if a file with the specified file name prefix exists in the destination directory. <p> Note Parquet files do not support the append mode. They support only the nonConflict mode.</p>	Yes	None
fieldDelimiter	The column delimiter used in the files to be written to HDFS. Make sure that you use the same delimiter as that in the Hive table. Otherwise, you cannot query data in the Hive table.	Yes (Not required if the fileType parameter is set to parquet)	None
compress	<p>The compression format of the files to be written to HDFS. By default, this parameter is left empty, that is, files are not compressed.</p> <p>For a text file, the GZIP and BZIP2 compression formats are supported. For an ORC file, the SNAPPY compression format is supported. To compress an ORC file, you must install SnappyCodec.</p>	No	None
encoding	The encoding format of the files to be written to HDFS.	No	None

Parameter	Description	Required	Default value
parquetSchema	<p>The schema of the files to be written to HDFS. This parameter is required only when the fileType parameter is set to parquet. Format:</p> <pre data-bbox="405 416 928 560">message messageType { required, dataType, columnName; ; }</pre> <p>Parameter description:</p> <ul style="list-style-type: none"> messageTypeName: the name of the MessageType object. required: specifies whether the field is required. We recommend that you set the parameter to optional for all fields. dataType: the type of the field. Valid values: BOOLEAN, INT32, INT64, INT96, FLOAT, DOUBLE, BINARY, and FIXED_LEN_BYTE_ARRAY. Set this parameter to BINARY if the field stores strings. <div data-bbox="405 945 928 1055" style="background-color: #e0f2f7; padding: 5px;"> <p> Note Each line, including the last one, must end with a semicolon (;).</p> </div> <p>Example:</p> <pre data-bbox="405 1120 928 1487">message m { optional int64 id; optional int64 date_id; optional binary datetimestring; optional int32 dspId; optional int32 advertiserId; optional int32 status; optional int64 bidding_req_num; optional int64 imp; optional int64 click_num; }</pre>	No	None

Parameter	Description	Required	Default value
hadoopConfig	<p>The advanced parameter settings of Hadoop, such as those related to high availability. The default resource group does not support configuring advanced Hadoop parameters related to the high availability feature.</p> <pre data-bbox="391 472 928 871"> "hadopConfig":{ "dfs.nameservices": "testDfs", "dfs.ha.namenodes.testDfs": "namenode1,namenode2", "dfs.namenode.rpc-address.youkuDfs.namenode1": "", "dfs.namenode.rpc-address.youkuDfs.namenode2": "", "dfs.client.failover.proxy.provider.testDfs": "org.apache.hadoop.hdfs.server.namenode.ha. ConfiguredFailoverProxyProvider" } </pre>	No	None

Parameter	Description	Required	Default value
dataxParquetMode	<p>The synchronization mode for Parquet files. If the dataxParquetMode parameter is set to fields, you can write data of complex types, such as ARRAY, MAP, and STRUCT. Valid values: fields and columns.</p> <p>If the dataxParquetMode parameter is set to fields, HDFS Writer supports HDFS over OSS. HDFS uses OSS as the storage service and HDFS Writer writes Parquet files to OSS. In this case, you can add the following OSS-related parameters in the hadoopConfig parameter:</p> <ul style="list-style-type: none"> fs.oss.accessKeyId: the AccessKey ID for connecting to OSS. fs.oss.accessKeySecret: the AccessKey secret for connecting to OSS. fs.oss.endpoint: the endpoint for connecting to OSS. <p>Example:</p> <pre> ` `` json "writer": { "name": "hdfswriter", "parameter": { "defaultFS": "oss://test-bucket", "fileType": "parquet", "path": "/datasets/oss_demo/kpt", "fileName": "test", "writeMode": "truncate", "compress": "SNAPPY", "encoding": "UTF-8", "hadoopConfig": { "fs.oss.accessKeyId": "the-access-id", "fs.oss.accessKeySecret": "the-access-key", "fs.oss.endpoint": "oss-cn-hangzhou.aliyuncs.com" }, "parquetSchema": "message test {\n required int64 id;\n optional binary name (UTF8);\n optional int64 gmt_create;\n required group map_col (MAP) {\n repeated group key_value {\n required binary key (UTF8);\n required binary value (UTF8);\n }\n }\n required group array_col (LIST) {\n repeated group list {\n required binary element (UTF8);\n }\n }\n required group struct_col {\n required int64 id;\n required binary name (UTF8);\n } \n}", "dataxParquetMode": "fields" } } ` `` </pre>	No	columns

Parameter	Description	If you set this parameter to <i>true</i> , you must also set the	Default value
haveKerberos	Specifies whether Kerberos authentication is required. Default value: <i>false</i> .	kerberosKeytabFilePath and kerberosPrincipal parameters.	<i>false</i>
kerberosKeytabFilePath	The absolute path of the keytab file for Kerberos authentication.	Required if the haveKerberos parameter is set to true	None
kerberosPrincipal	<p>The Kerberos principal to which Kerberos can assign tickets. Example: <code>****/hadoopclient@**.***</code>.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p>? Note The absolute path of the keytab file is required for Kerberos authentication. Therefore, you can configure Kerberos authentication only on a custom resource group. Example:</p> <pre style="background-color: #f5f5f5; padding: 5px;">"haveKerberos":true, "kerberosKeytabFilePath":"/opt/datax/**.keytab", "kerberosPrincipal": "**/hadoopclient@**.***"</pre> </div>	Required if the haveKerberos parameter is set to true	None

Configure HDFS Writer by using the codeless UI

The codeless UI is not supported for HDFS Writer.

Configure HDFS Writer by using the code editor

In the following code, a node is configured to write files to HDFS. For more information about the parameters, see the preceding parameter description.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "hdfs", // The writer type.
      "parameter": {
        "path": "", // The directory in HDFS to which the files are written.
        "fileName": "", // The name prefix of the files to be written to HDFS.
        "compress": "", // The compression format of the files.
        "datasource": "", // The connection name.
        "column": [
          {
            "name": "col1", // The name of the column.
            "type": "string" // The data type of the column.
```

```

        type : string // The data type of the column.
    },
    {
        "name": "col2",
        "type": "int"
    },
    {
        "name": "col3",
        "type": "double"
    },
    {
        "name": "col4",
        "type": "boolean"
    },
    {
        "name": "col5",
        "type": "date"
    }
    ],
    "writeMode": "", // The write mode.
    "fieldDelimiter": ",", // The column delimiter.
    "encoding": "", // The encoding format.
    "fileType": "text" // The file format.
},
"name": "Writer",
"category": "writer"
}
],
"setting": {
    "errorLimit": {
        "record": "" // The maximum number of dirty data records allowed.
    },
    "speed": {
        "concurrent": 3, // The maximum number of concurrent threads.
        "throttle": false // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    }
},
"order": {
    "hops": [
        {
            "from": "Reader",
            "to": "Writer"
        }
    ]
}
}
}

```

2.4.5.4.9. Configure MaxCompute Writer

This topic describes the data types and parameters supported by MaxCompute Writer and how to configure it by using the codeless UI and code editor.

MaxCompute Writer is designed for developers to insert data to or update data in MaxCompute. MaxCompute Writer is suitable for importing data at the GB or TB level to MaxCompute.

 **Note** You must configure a connection before you configure MaxCompute Writer.

Based on the specified information such as the source project, table, partition, and field, MaxCompute Writer writes data to MaxCompute by using Tunnel.

Data types

The following table lists the data types supported by MaxCompute Writer.

Category	MaxCompute data type
Integer	BIGINT
Floating point	DOUBLE and DECIMAL
String	STRING
Date and time	DATETIME
Boolean	BOOLEAN

Parameters

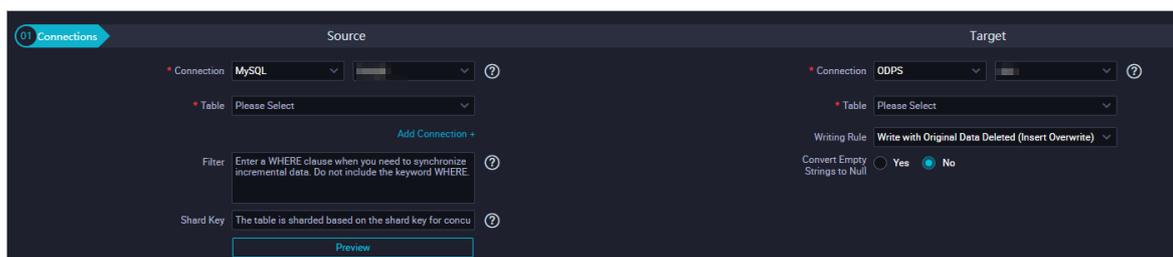
Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the destination table. The name is not case-sensitive. You can specify only one table as the destination table.	Yes	None
partition	<p>The partition to which data is written. The last-level partition must be specified. For example, if you want to write data to a three-level partitioned table, set the partition parameter to a value that contains the third-level partition information, for example, <code>pt=20150101, type=1, biz=2</code>.</p> <ul style="list-style-type: none"> To write data to a non-partitioned table, do not set this parameter. The data is directly written to the destination table. MaxCompute Writer does not support writing data based on the partition route. To write data to a partitioned table, make sure that data is written to the last-level partition. 	Required only for writing data to a partitioned table	None

Parameter	Description	Required	Default value
column	<p>The columns in the destination table to which data is written. To write data to all the columns in the destination table, set the value to an asterisk (*), for example, <code>"column":["*"]</code>. Set the value to the specified columns if data is written to only some of the columns in the destination table. Separate the columns with commas (,). Example: <code>"column":["id","name"]</code>.</p> <ul style="list-style-type: none"> MaxCompute Writer can filter columns and change the order of columns. For example, a MaxCompute table has three columns: a, b, and c. If you want to write data only to column c and column b, you can set the column parameter in the format <code>"column":["c","b"]</code>. During data synchronization, column a is automatically set to null. The column parameter must explicitly specify a set of columns to which data is written. The parameter cannot be left empty. 	Yes	None
truncate	<p>To ensure the idempotence of write operations, set the truncate parameter in the format <code>"truncate":"true"</code>. When a failed sync node is rerun due to a write failure, MaxCompute Writer deletes the data that has been written before it imports the source data again. This ensures that the same data is written for each rerun.</p> <p>MaxCompute Writer uses MaxCompute SQL to delete data. MaxCompute SQL cannot ensure the atomicity. Therefore, the truncation operation is not an atomic operation. Conflicts may occur when concurrent nodes delete data from the same table or partition.</p> <p>To avoid this issue, we recommend that you do not run concurrent DDL nodes to write data to the same partition. You can create different partitions for nodes that need to run concurrently.</p>	Yes	None

Configure MaxCompute Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.



Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.

Parameter	Description
Partition Key Column	<p>To write data to all the columns in the destination table, set "column":["*"] for the column parameter in the preceding parameter description. The partition parameter allows you to use wildcards and specify one or more partitions.</p> <ul style="list-style-type: none"> ◦ "partition":"pt=20140501/ds=*" specifies that data is written to all ds partitions with pt=20140501. ◦ "partition":"pt=top?" specifies that data is written to the partitions with pt=top and pt=to. <p>You can specify the partition key columns to which data is written. Assume that the partition key column of a MaxCompute table is pt=\${bdp.system.bizdate}. You can configure the column to which data is written to pt. Ignore it if the column is marked as unidentified.</p> <ul style="list-style-type: none"> ◦ To write data to all partitions, enter pt=*. ◦ To write data to some of the partitions, specify the corresponding dates.
Writing Rule	<ul style="list-style-type: none"> ◦ Write with Original Data Deleted (Insert Overwrite): All data in the table or partition is deleted before data import. This rule is equivalent to the <code>INSERT OVERWRITE</code> statement. ◦ Write with Original Data Retained (Insert Into): No data is deleted before data import. New data is always appended upon each run. This rule is equivalent to the <code>INSERT INTO</code> statement. <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> ◦ MaxCompute Reader reads data by using Tunnel. Sync nodes do not support data filtering. Instead, they must read all the data in a specific table or partition. ◦ MaxCompute Writer writes data by using Tunnel instead of the <code>INSERT INTO</code> statement. You can view the complete data in the destination table only after a sync node is run. Pay attention to the node dependencies. </div>
Convert Empty Strings to Null	Specifies whether to convert empty strings to null. Default value: <i>No</i> .

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure MaxCompute Writer by using the code editor

In the following code, a node is configured to write data to a MaxCompute project. For more information about the parameters, see the preceding parameter description.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"odps",// The writer type.
      "parameter":{
        "partition":"","// The partition information.
        "truncate":true,// The write rule.
        "compress":false,// Specifies whether to enable compression.
        "datasource":"odps_first",// The connection name.
        "column":[// The columns to which data is written.
          "id",
          "name",
          "age",
          "sex",
          "salary",
          "interest"
        ],
        "emptyAsNull":false,// Specifies whether to convert empty strings to null.
        "table":"","// The name of the destination table.
      },
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0"// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1 // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

Additional instructions

- Column filter

By configuring MaxCompute Writer, you can perform operations that MaxCompute does not support, for example, filter columns, reorder columns, and set empty fields to null. To write data to all the columns in the destination table, set the column parameter in the format `"column":["*"]`.

For example, a MaxCompute table has three columns: a, b, and c. If you want to write data only to column c and column b, you can set the column parameter in the format `"column":["c","b"]`. The first column and the second column of the source data are written to column c and column b in the MaxCompute table respectively. During data synchronization, column a is automatically set to null.

- Column configuration error handling

To avoid losing the data of redundant columns and ensure high data reliability, MaxCompute Writer returns an error message if the number of columns to be written is more than that in the destination table. For example, if a MaxCompute table contains columns a, b, and c, MaxCompute Writer returns an error message if more than three columns are to be written to the table.

- Partition configuration

MaxCompute Writer can write data only to the last-level partition, and cannot write data to the specified partition based on a field. To write data to a partitioned table, specify the last-level partition. For example, if you want to write data to a three-level partitioned table, set the partition parameter to a value that contains the third-level partition information, for example, `pt=20150101.type=1.biz=2`. The data cannot be written if you set the partition parameter to `pt=20150101.type=1` or `pt=20150101`.

- Node rerunning

To ensure the idempotence of write operations, set the `truncate` parameter to true. When a failed sync node is rerun due to a write failure, MaxCompute Writer deletes the data that has been written before it imports the source data again. This ensures that the same data is written for each rerun. If a sync node is interrupted due to other exceptions, the data cannot be rolled back and the node cannot be rerun automatically. You can ensure the idempotence of write operations and the data integrity by setting the `truncate` parameter to true.

 **Note** If the `truncate` parameter is set to true, all data of the specified partition or table is deleted before a rerun. Exercise caution when you set this parameter to true.

2.4.5.4.10. Configure Memcache Writer

This topic describes the data types and parameters supported by Memcache Writer and how to configure it by using the code editor.

ApsaraDB for Memcache is a distributed in-memory database service with high performance, reliability, and scalability. Based on the Apsara distributed operating system and high-performance storage technologies, ApsaraDB for Memcache provides a complete database solution with hot standby, fault recovery, business monitoring, and data migration features.

ApsaraDB for Memcache is immediately available after an instance is created. It relieves the load on databases from dynamic websites and applications by caching data in the memory and therefore improves the response speed of websites and applications.

Same as on-premises Memcached databases, ApsaraDB for Memcache databases are compatible with the Memcached protocol. ApsaraDB for Memcache databases can be directly used in your environments. The difference is that the data, hardware infrastructure, network security, and system maintenance services used by ApsaraDB for Memcache databases are all deployed in the cloud. These services are billed based on the pay-as-you-go billing method.

Memcache Writer writes data to ApsaraDB for Memcache databases based on the Memcached protocol.

Memcache Writer writes data only in text format. The method of converting data types varies based on the format of writing data.

- **text**: Memcache Writer uses the specified column delimiter to serialize source data to a string.
- **binary**: This format is not supported.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
writeMode	<p>The write mode. Valid values:</p> <ul style="list-style-type: none"> • <i>set</i>: stores the source data. • <i>add</i>: stores the source data only when its key does not exist in the destination ApsaraDB for Memcache database. This mode is not supported now. • <i>replace</i>: uses the source data to replace the data record with the same key in the destination ApsaraDB for Memcache database. This mode is not supported now. • <i>append</i>: adds the value of the source data to the end of the value of an existing data record with the same key in the destination ApsaraDB for Memcache database, but does not update the expiration time of the existing data record. This mode is not supported now. • <i>prepend</i>: adds the value of the source data to the beginning of the value of an existing data record with the same key in the destination ApsaraDB for Memcache database, but does not update the expiration time of the existing data record. This mode is not supported now. 	Yes	None
writeFormat	<p>The format in which Memcache Writer writes the source data. Currently, only the text format is supported.</p> <p>text: serializes the source data to the text format. Memcache Writer uses the first column of the source data as the key and serializes the subsequent columns to the value by using the specified delimiter. Then, Memcache Writer writes the key-value pair to ApsaraDB for Memcache.</p> <p>Assume that the following source data exists:</p> <pre> ID NAME COUNT --- :--- :--- 23 "CDP" 100 </pre> <p>If you set the column delimiter to a backslash and a caret (\^), data is written to ApsaraDB for Memcache in the following format:</p> <pre> KEY (OCS) VALUE(OCS) :----- :----- 23 CDP\^100 </pre>	No	None

Parameter	Description	Required	Default value
expireTime	<p>The expiration time of the source data to be cached in ApsaraDB for Memcache. ApsaraDB for Memcache supports the following two types of expiration time:</p> <ul style="list-style-type: none"> • unixtime: the UNIX timestamp, indicating a specific time point in the future when the data expires. The UNIX timestamp represents the number of seconds that have elapsed since 00:00:00 on January 1, 1970. • seconds: the relative time in seconds starting from the current time point. It specifies the period during which data is valid. <div style="background-color: #e0f2f7; padding: 5px; margin-top: 10px;"> <p> Note If the specified time exceeds 30 days, the server identifies the time as the UNIX timestamp.</p> </div>	No	0, indicating that the data never expires
batchSize	<p>The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the ApsaraDB for Memcache database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.</p>	No	1,024

Configure Memcache Writer by using the codeless UI

The codeless UI is not supported for Memcache Writer.

Configure Memcache Writer by using the code editor

In the following code, a node is configured to write data to an ApsaraDB for Memcache database.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"ocs",// The writer type.
      "parameter":{
        "writeFormat":"text",// The format in which Memcache Writer writes the source data.
        "expireTime":1000,// The expiration time of the source data to be cached in ApsaraDB for Memcache.
        "indexes":0,
        "datasource":"","// The connection name.
        "writeMode":"set",// The write mode.
        "batchSize":"256"// The number of data records to write at a time.
      },
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0" // The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1 // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

2.4.5.4.11. Configure MongoDB Writer

This topic describes the data types and parameters supported by MongoDB Writer and how to configure it by using the code editor.

MongoDB Writer connects to a remote MongoDB database by using the Java client named MongoClient and writes data to the database. The latest version of MongoDB has improved the locking feature from database locks to document locks. The powerful index functionalities of MongoDB enable MongoDB Writer to efficiently write data to MongoDB databases. If you want to update data, specify the primary key.

Note

- You must configure a connection before you configure MongoDB Writer.
- If you use ApsaraDB for MongoDB, the MongoDB database has a root account by default.
- For security concerns, Data Integration only supports access to a MongoDB database by using a MongoDB database account. When you add a MongoDB connection, do not use the root account for access.

MongoDB Writer obtains data from a Data Integration reader, and converts the data types to those supported by MongoDB. Data Integration does not support arrays. MongoDB supports arrays and the array index is useful.

To use MongoDB arrays, you can convert strings to MongoDB arrays by configuring a parameter and write the arrays to a MongoDB database.

Data types

MongoDB Writer supports most MongoDB data types. Make sure that your data types are supported.

The following table lists the data types supported by MongoDB Writer.

Category	MongoDB data type
Integer	INT and LONG
Floating point	DOUBLE
String	STRING and ARRAY
Date and time	DATE
Boolean	BOOL
Binary	BYTES

Note When data of the DATE type is written to a MongoDB database, the type of the data is converted to DATETIME.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
collectionName	The name of the MongoDB collection.	Yes	None

Parameter	Description	Required	Default value
column	<p>The columns in MongoDB.</p> <ul style="list-style-type: none">• name: the name of the column.• type: the data type of the column.• splitter: the delimiter. Specify this field only when you want to convert the string to an array. The string is split based on the specified delimiter, and the split strings are saved in a MongoDB array.	Yes	None
writeMode	<p>Specifies whether to overwrite data.</p> <ul style="list-style-type: none">• isReplace: If you set this parameter to true, MongoDB Writer overwrites the data in the destination table with the same primary key. If you set this parameter to false, the data is not overwritten.• replaceKey: the primary key for each record. Data is overwritten based on this primary key. The primary key must be unique.	No	None

Parameter	Description	Required	Default value
preSql	<p>The action to perform before the sync node is run. For example, you can clear outdated data before data synchronization. If the preSql parameter is left empty, no action is performed before data synchronization. Make sure that the value of the preSql parameter complies with the JSON syntax. The format requirements for the preSql parameter are as follows:</p> <ul style="list-style-type: none"> • Configure the type field to specify the action type. Valid values: drop and remove. Example: <code>"preSql":{"type":"remove"}</code> . <ul style="list-style-type: none"> ◦ <i>drop</i>: deletes the collection specified by the collectionName parameter and the data in the collection. ◦ <i>remove</i>: deletes data based on conditions. ◦ <i>json</i>: the conditions for deleting data. Example: <code>"preSql":{"type":"remove","json":{"operationTime":{"\$gte":{"\$date":{"\$last day":{"T00:00:00.424+0800'}}}}}}</code> . In the preceding JSON string, <code>last day</code> is a scheduling parameter of DataWorks. The format is <code>yy-mm-dd</code> . You can use comparison operators (such as <code>\$gt</code>, <code>\$lt</code>, <code>\$gte</code>, and <code>\$lte</code>), logical operators (such as <code>\$and</code> and <code>\$or</code>), and functions (such as <code>max</code>, <code>min</code>, <code>sum</code>, <code>avg</code>, and <code>ISODate</code>) supported by MongoDB as needed. For more information, see the MongoDB query syntax. <p>Data Integration uses the following standard MongoDB API to query and delete the specified data:</p> <pre>query=(BasicDBObject) com.mongodb.util.JSON.parse(json); col.deleteMany(query);</pre> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note If you want to delete data based on conditions, we recommend that you specify the conditions in JSON format preferentially.</p> </div> <ul style="list-style-type: none"> ◦ <i>item</i>: the name, condition, and value for filtering data. Example: <code>"preSql":{"type":"remove","item":{"name":"nv","value":"100","condition":{"\$gt":{"name":"pid","value":"10"}}}}</code> . <p>Data Integration sets query conditions based on the value of the item field and deletes data by using the standard MongoDB API. Example: <code>col.deleteMany(query);</code> .</p> <ul style="list-style-type: none"> • If the value of the preSql parameter cannot be recognized, no action is performed. 	No	None

Configure MongoDB Writer by using the codeless UI

The codeless UI is not supported for MongoDB Writer.

Configure MongoDB Writer by using the code editor

In the following code, a node is configured to write data to a MongoDB database. For more information about the parameters, see the preceding parameter description.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "mongodb", // The writer type.
      "parameter": {
        "datasource": "", // The connection name.
        "column": [
          {
            "name": "_id", // The name of the column to which data is written.
            "type": "ObjectId" // The data type of the column to which data is written. If the replacekey parameter is
            set to _id, set the type parameter to ObjectId. If you set the type parameter to String, the data cannot be overwritt
            en.
          },
          {
            "name": "age",
            "type": "int"
          },
          {
            "name": "id",
            "type": "long"
          },
          {
            "name": "wealth",
            "type": "double"
          },
          {
            "name": "hobby",
            "type": "array",
            "splitter": ""
          },
          {
            "name": "valid",
            "type": "boolean"
          },
          {
            "name": "date_of_join",
            "format": "yyyy-MM-dd HH:mm:ss",
            "type": "date"
          }
        ],
        "writeMode": { // The write mode.
          "isReplace": "true",
          "replaceKey": "_id"
        },
        "collectionName": "datax_test" // The name of the MongoDB collection.
      },
      "name": "Writer"
    }
  ]
}
```

```

    name: writer,
    "category": "writer"
  }
],
"setting": {
  "errorLimit": { // The maximum number of dirty data records allowed.
    "record": "0"
  },
  "speed": {
    "jvmOption": "-Xms1024m -Xmx1024m",
    "throttle": true, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent": 1, // The maximum number of concurrent threads.
    "mbps": "1" // The maximum transmission rate.
  }
},
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}
}

```

2.4.5.4.12. Configure MySQL Writer

This topic describes the data types and parameters supported by MySQL Writer and how to configure it by using the codeless UI and code editor.

MySQL Writer allows you to write data to tables stored in MySQL databases. MySQL Writer connects to a remote MySQL database by using JDBC, and executes the `INSERT INTO` or `REPLACE INTO` statement to write data to the MySQL database. MySQL uses the InnoDB engine so that data is written to the database in batches.

Note

- You must configure a connection before you configure MySQL Writer.
- MySQL Writer does not support MySQL 8.0 or later.

MySQL Writer can be used as a data migration tool by users such as DBAs. MySQL Writer obtains data from a Data Integration reader, and writes the data to the destination database based on value of the `writeMode` parameter.



Note A `sync` node that uses MySQL Writer must have at least the permission to execute the `INSERT INTO` or `REPLACE INTO` statement. Whether other permissions are required depends on the SQL statements specified in the `preSql` and `postSql` parameters when you configure the node.

Data types

MySQL Writer supports most MySQL data types. Make sure that your data types are supported.

The following table lists the data types supported by MySQL Writer.

Category	MySQL data type
Integer	INT, TINYINT, SMALLINT, MEDIUMINT, BIGINT, and YEAR
Floating point	FLOAT, DOUBLE, and DECIMAL
String	VARCHAR, CHAR, TINYTEXT, TEXT, MEDIUMTEXT, and LONGTEXT
Date and time	DATE, DATETIME, TIMESTAMP, and TIME
Boolean	BOOL
Binary	TINYBLOB, MEDIUMBLOB, BLOB, LONGBLOB, and VARBINARY

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the destination table.	Yes	None
writeMode	<p>The write mode. Valid values:</p> <ul style="list-style-type: none"> <i>insert into</i>: If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows and is regarded as dirty data. <i>on duplicate key update</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, specified fields in original rows are updated. <i>replace into</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, original rows are deleted and new rows are inserted. This means that all the field values of the original rows are replaced. 	No	<i>insert</i>
column	The columns in the destination table to which data is written. Separate the columns with commas (,), for example, "column": ["id","name","age"] . To write data to all the columns in the destination table, set the value to an asterisk (*), for example, "column":["*"] .	Yes	None
preSql	<p>The SQL statement to execute before the sync node is run. For example, you can clear outdated data before data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.</p> <p> Note If you specify multiple SQL statements in the code editor, the system may not execute them in the same transaction.</p>	No	None

Parameter	Description	Required	Default value
postSql	<p>The SQL statement to execute after the sync node is run. For example, you can add a timestamp after data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>? Note If you specify multiple SQL statements in the code editor, the system may not execute them in the same transaction.</p> </div>	No	None
batchSize	<p>The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the MySQL database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.</p>	No	1,024

Configure MySQL Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Statement Run Before Writing	The preSql parameter in the preceding parameter description. Enter an SQL statement to execute before the sync node is run.
Statement Run After Writing	The postSql parameter in the preceding parameter description. Enter an SQL statement to execute after the sync node is run.
Solution to Primary Key Violation	The writeMode parameter in the preceding parameter description. Select the required write mode.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure MySQL Writer by using the code editor

In the following code, a node is configured to write data to a MySQL database. For more information about the parameters, see the preceding parameter description.

```

{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"mysql",// The writer type.
      "parameter":{
        "postSql":[// The SQL statement to execute after the sync node is run.
          "datasource":"","// The connection name.
          "column":[// The columns to which data is written.
            "id",
            "value"
          ],
          "writeMode":"insert",// The write mode.
          "batchSize":1024,// The number of data records to write at a time.
          "table":"","// The name of the destination table.
          "preSql":[// The SQL statement to execute before the sync node is run.
        ],
        "name":"Writer",
        "category":"writer"
      }
    }
  ],
  "setting":{
    "errorLimit":{// The maximum number of dirty data records allowed.
      "record":"0"
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling.
      "concurrent":1 // The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}

```

2.4.5.4.13. Configure Oracle Writer

This topic describes the data types and parameters supported by Oracle Writer and how to configure it by using the codeless UI and code editor.

Oracle Writer allows you to write data to tables stored in primary Oracle databases. Oracle Writer connects to a remote Oracle database by using JDBC, and executes an `INSERT INTO` statement to write data to the Oracle database.

 **Note** You must configure a connection before you configure Oracle Writer.

Oracle Writer is designed for ETL developers to import data from data warehouses to Oracle databases. Oracle Writer can also be used as a data migration tool by users such as DBAs.

Oracle Writer obtains data from a Data Integration reader, connects to a remote Oracle database by using JDBC, and then executes an SQL statement to write data to the Oracle database.

Data types

Oracle Writer supports most Oracle data types. Make sure that your data types are supported.

The following table lists the data types supported by Oracle Writer.

Category	Oracle data type
Integer	NUMBER, ROWID, INTEGER, INT, and SMALLINT
Floating point	NUMERIC, DECIMAL, FLOAT, DOUBLE PRECISION, and REAL
String	LONG, CHAR, NCHAR, VARCHAR, VARCHAR2, NVARCHAR2, CLOB, NCLOB, CHARACTER, CHARACTER VARYING, CHAR VARYING, NATIONAL CHARACTER, NATIONAL CHAR, NATIONAL CHARACTER VARYING, NATIONAL CHAR VARYING, and NCHAR VARYING
Date and time	TIMESTAMP and DATE
Boolean	BIT and BOOLEAN
Binary	BLOB, BFILE, RAW, and LONG RAW

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the destination table.	Yes	None
writeMode	<p>The write mode. Valid values: <i>insert into</i>, <i>on duplicate key update</i>, and <i>replace into</i>.</p> <ul style="list-style-type: none"> <i>insert into</i>: If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows and is regarded as dirty data. <i>on duplicate key update</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, specified fields in original rows are updated. <i>replace into</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, original rows are deleted and new rows are inserted. This means that all the field values of the original rows are replaced. 	No	<i>insert</i>

Parameter	Description	Required	Default value
column	The columns in the destination table to which data is written. Separate the columns with commas (,), for example, <code>"column": ["id","name","age"]</code> . To write data to all the columns in the destination table, set the value to an asterisk (*), for example, <code>"column":["*"]</code> .	Yes	None
preSql	The SQL statement to execute before the sync node is run. For example, you can clear outdated data before data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
postSql	The SQL statement to execute after the sync node is run. For example, you can add a timestamp after data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the Oracle database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,024

Configure Oracle Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Statement Run Before Writing	The preSql parameter in the preceding parameter description. Enter an SQL statement to execute before the sync node is run.
Statement Run After Writing	The postSql parameter in the preceding parameter description. Enter an SQL statement to execute after the sync node is run.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.

GUI element	Description
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure Oracle Writer by using the code editor

In the following code, a node is configured to write data to an Oracle database.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "oracle", // The writer type.
      "parameter": {
        "postSql": [], // The SQL statement to execute after the sync node is run.
        "datasource": "",
        "session": [], // The settings of the session to the database.
        "column": [], // The columns to which data is written.
          "id",
          "name"
        ],
        "encoding": "UTF-8", // The encoding format.
        "batchSize": 1024, // The number of data records to write at a time.
        "table": "", // The name of the destination table.
        "preSql": [] // The SQL statement to execute before the sync node is run.
      },
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent": 1 // The maximum number of concurrent threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.4.14. Configure OSS Writer

This topic describes the data types and parameters supported by OSS Writer and how to configure it by using the codeless UI and code editor.

OSS Writer allows you to write one or more CSV-like files to OSS.

 **Note** You must configure a connection before you configure OSS Writer.

OSS Writer can write files that store logical two-dimensional tables, such as CSV files that store text data, to OSS.

OSS Writer allows you to convert data obtained from a Data Integration reader to files and write the files to OSS. The OSS files store unstructured data only. OSS Writer supports the following features:

- Writes only files that store text data. The text data must be logical two-dimensional tables.
- Writes CSV-like files with custom delimiters.
- Uses concurrent threads to write files. Each thread writes a file.
- Supports file rotation. OSS Writer can write data to another file when the size of the current file exceeds a specific value. OSS Writer can also write data to another file when the number of rows in the current file exceeds a specific value.

OSS Writer does not support the following features:

- Uses concurrent threads to write a single file.
- Distinguishes between data types. OSS does not distinguish between data types. Therefore, OSS Writer writes all data as strings to files in OSS.

The following table lists the data types supported by OSS Writer.

Category	OSS data type
Integer	LONG
Floating point	DOUBLE
String	STRING
Boolean	BOOLEAN
Date and time	DATE

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
object	<p>The name prefix of the files to be written to OSS as objects. OSS simulates the directory effect by adding separators to object names. You can set the object parameter based on the following rules:</p> <ul style="list-style-type: none"> • <code>"object": "datax"</code> : The names of the files start with datax, which is followed by a random string as the suffix. • <code>"object": "cdo/datax"</code> : The names of the files start with /cdo/datax, which is followed by a random string as the suffix. OSS uses backslashes (/) in objects to simulate the directory effect. <p>If you do not want to add a random universally unique identifier (UUID) as the suffix, we recommend that you set the writeSingleObject parameter to <code>true</code>.</p>	Yes	None

Parameter	Description	Required	Default value
writeMode	<p>The mode in which OSS Writer writes the files. Valid values:</p> <ul style="list-style-type: none"> <i>truncate</i>: deletes all existing objects with the specified object name prefix before writing files to OSS. For example, if you set the object parameter to <code>abc</code>, all objects whose names start with <code>abc</code> are deleted. <i>append</i>: writes all files and ensures that the actual file names do not conflict with those of existing objects by suffixing the file names with random UUIDs. For example, if you set the object parameter to <code>DI</code>, the actual names of the files written to OSS are in the following format: <code>DI_****_****_****</code>. <i>nonConflict</i>: returns an error message if an object with the specified object name exists. For example, if you set the <code>object</code> parameter to <code>abc</code> and the object named <code>abc123</code> exists, an error message is returned. 	Yes	None
fileFormat	<p>The format in which the files are written to OSS. Valid values: <code>csv</code> and <code>text</code>.</p> <ul style="list-style-type: none"> If a file is written as a CSV file, the file strictly follows CSV specifications. If the data in the file contains the column delimiter, the column delimiter is escaped by using double quotation marks ("<code>"</code>"). If a file is written as a text file, the data in the file is separated with the column delimiter. If the data in the file contains the column delimiter, the column delimiter is not escaped. 	No	<i>text</i>
fieldDelimiter	The column delimiter that is used in the files to be written to OSS.	No	,
encoding	The encoding format of the files to be written to OSS.	No	<i>utf-8</i>
nullFormat	The string that represents null. No standard strings can represent null in text files. Therefore, Data Integration provides the <code>nullFormat</code> parameter to define which string represents a null pointer. For example, if you specify <code>nullFormat="null"</code> , Data Integration considers null as a null pointer.	No	None
header (advanced parameter, which cannot be set on the codeless UI)	The table header in the files to be written to OSS, for example, <code>['id','name','age']</code> .	No	None

Parameter	Description	Required	Default value
maxFileSize (advanced parameter, which cannot be set on the codeless UI)	<p>The maximum size of a single file that can be written to OSS. Default value: 100000. Unit: MB. File rotation based on this maximum size is similar to log rotation of Log4j. When a file is uploaded to OSS in multiple parts, the minimum size of a part is 10 MB. This size is the minimum granularity for file rotation. That is, if you set the maxFileSize parameter to less than 10 MB, the minimum size of a file is still 10 MB. Each call of the InitiateMultipartUploadRequest operation supports writing up to 10,000 parts.</p> <p>If file rotation occurs, suffixes, such as <code>_1</code>, <code>_2</code>, and <code>_3</code>, are appended to the new file names that consist of file name prefixes and random UUIDs.</p>	No	100,000MB
suffix (advanced parameter, which cannot be set on the codeless UI)	<p>The file name extension of the files to be written to OSS. For example, if you set the suffix parameter to <code>.csv</code>, the final name of a file written to OSS is in the format <code>fileName****.csv</code>.</p>	No	None

Configure OSS Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Object Name Prefix	The object parameter in the preceding parameter description. Enter the path of the directory for storing the files. Do not include the name of the OSS bucket in the path.
File Type	The fileFormat parameter in the preceding parameter description. Valid values: <code>csv</code> and <code>text</code> .
Field Delimiter	The fieldDelimiter parameter in the preceding parameter description. The default delimiter is comma (<code>,</code>).
Encoding	The encoding parameter in the preceding parameter description. Default value: <code>UTF-8</code> .
Null String	The nullFormat parameter in the preceding parameter description. Enter a string that represents null. If the data in the source data store contains the string, the string is replaced with null.

Parameter	Description
Time Format	The format in which the data of the DATE type is serialized in an object, for example, <code>"dateFormat": "yyyy-MM-dd"</code> .
Solution to Duplicate Prefixes	The solution to take when a prefix conflict occurs. If an object with the specified name prefix exists, replace the object with the new object, insert the new object, or return an error message.

2. Configure field mapping. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure OSS Writer by using the code editor

In the following code, a node is configured to write files to OSS. For more information about the parameters, see the preceding parameter description.

```
{
  "type": "job",
  "version": "2.0",
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "oss", // The writer type.
      "parameter": {
        "nullFormat": "", // The string that represents null.
        "dateFormat": "", // The format in which the data of the DATE type is serialized in an object.
        "datasource": "", // The connection name.
        "writeMode": "", // The write mode.
        "encoding": "", // The encoding format.
        "fieldDelimiter": ",", // The column delimiter.
        "fileFormat": "", // The format in which the files are written to OSS.
        "object": "" // The name prefix of the files to be written to OSS as objects.
      },
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent": 1 // The maximum number of concurrent threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}
```

2.4.5.4.15. Configure PostgreSQL Writer

This topic describes the data types and parameters supported by PostgreSQL Writer and how to configure it by using the codeless UI and code editor.

PostgreSQL Writer allows you to write data to a PostgreSQL database. PostgreSQL Writer connects to a remote PostgreSQL database by using JDBC, and executes an SQL statement to write data to the PostgreSQL database.

 **Note** You must configure a connection before you configure PostgreSQL Writer.

- PostgreSQL Writer generates the SQL statement based on the table, column, and where parameters that you specified, and sends the generated SQL statement to the PostgreSQL database.
- If you specify the querySql parameter, PostgreSQL Writer directly sends the value of this parameter to the PostgreSQL database.

Data types

PostgreSQL Writer supports most PostgreSQL data types. Make sure that your data types are supported.

The following table lists the data types supported by PostgreSQL Writer.

Data Integration data type	PostgreSQL data type
LONG	BIGINT, BIGSERIAL, INTEGER, SMALLINT, and SERIAL
DOUBLE	DOUBLE, PRECISION, MONEY, NUMERIC, and REAL
STRING	VARCHAR, CHAR, TEXT, BIT, and INET
DATE	DATE, TIME, and TIMESTAMP
BOOLEAN	BOOL
BYTES	BYTEA

 **Note**

- Data types that are not listed in the table are not supported.
- You can convert the MONEY, INET, and BIT types by using syntax such as `a_inet::varchar`.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the destination table.	Yes	None
writeMode	<p>The write mode. Valid values: insert and copy.</p> <ul style="list-style-type: none"> • <i>insert</i>: executes the <code>INSERT INTO</code> statement to write data to the PostgreSQL database. If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows and is regarded as dirty data. We recommend that you select the insert mode. • <i>copy</i>: copies data between tables and the standard input or output file. Data Integration supports the <code>COPY FROM</code> command, which allows you to copy data from files to tables. We recommend that you try this mode when performance issues occur. 	No	<i>insert</i>

Parameter	Description	Required	Default value
column	The columns in the destination table to which data is written. Separate the columns with commas (,), for example, <code>"column": ["id","name","age"]</code> . To write data to all the columns in the destination table, set the value to an asterisk (*), for example, <code>"column":["*"]</code> .	Yes	None
preSql	The SQL statement to execute before the sync node is run. For example, you can clear outdated data before data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
postSql	The SQL statement to execute after the sync node is run. For example, you can add a timestamp after data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the PostgreSQL database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,024

Parameter	Description	Required	Default value
pgType	<p>The PostgreSQL configuration for converting data types. Valid values: bigint[], double[], text[], jsonb, and json. Example:</p> <pre> { "job": { "content": { "reader": {...}, "writer": { "parameter": { "column": [// The columns in the destination table to which data is written. "bigint_arr", "double_arr", "text_arr", "jsonb_obj", "json_obj"], "pgType": { // The PostgreSQL configuration for converting data types. In each key-value pair, the key specifies the name of a field in the destination table, and the value specifies the data type of the field. "bigint_arr": "bigint[]", "double_arr": "double[]", "text_arr": "text[]", "jsonb_obj": "jsonb", "json_obj": "json" } } } } } } </pre>	No	None

Configure PostgreSQL Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.

Parameter	Description
Table	The table parameter in the preceding parameter description.
Statement Run Before Writing	The preSql parameter in the preceding parameter description. Enter an SQL statement to execute before the sync node is run.
Statement Run After Writing	The postSql parameter in the preceding parameter description. Enter an SQL statement to execute after the sync node is run.
Write Method	The writeMode parameter in the preceding parameter description. Valid values: <i>Insert</i> and <i>Copy</i> .

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

Button	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure PostgreSQL Writer by using the code editor

In the following code, a node is configured to write data to a PostgreSQL database. For more information about the parameters, see the preceding parameter description.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "postgresql", // The writer type.
      "parameter": {
        "postSql": [], // The SQL statement to execute after the sync node is run.
        "datasource": "", // The connection name.
        "col1",
        "col2"
      },
      "table": "", // The name of the destination table.
      "preSql": [] // The SQL statement to execute before the sync node is run.
    },
    {
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "0" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent": "1" // The maximum number of concurrent threads.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.4.16. Configure Redis Writer

Redis Writer is a writer that is developed based on the Data Integration framework. It can be used to import data from data stores such as data warehouses to Redis databases.

Redis is a network-enabled key-value storage system that is either in-memory or permanent. It supports logs and delivers high performance. It can be used as a database, cache, and message broker. Redis supports diverse data types for values, including STRING, LIST, SET, ZSET (sorted set), and HASH.

Redis Writer interacts with a Redis server by using Jedis. As a preferred Java client development kit provided by Redis, Jedis supports almost all the features of Redis.

Note

- You must configure a connection before you configure Redis Writer.
- If you write values of the LIST type to Redis by using Redis Writer, the result of rerunning a sync node is not idempotent. If the data type of the values is LIST, you must manually clear the corresponding data on Redis when you rerun a sync node.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
keyIndexes	The columns used as the key. The index of the first column is 0. For example, if you want to set the first and second columns of the source data as the key, set the keyIndexes parameter to [0,1]. Note After you specify the keyIndexes parameter, Redis Writer specifies the remaining columns as the value. If you do not want to synchronize all the columns, filter columns when you configure the reader.	Yes	None
keyFieldDelimiter	The delimiter used to separate keys when data is written to Redis. Example: key=key1\u0001id. If multiple keys need to be concatenated, this parameter is required. If only one key exists, this parameter is not required.	No	\u0001
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the Redis database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,000
expireTime	The expiration time of the values to be cached in Redis. Unit: seconds. The data is valid permanently if you do not specify this parameter. <ul style="list-style-type: none"> • <i>seconds</i>: the relative time in seconds starting from the current time point. It specifies the time range during which data is valid. • <i>unixtime</i>: the UNIX timestamp, indicating that data is invalid at a specific time point in the future. The UNIX timestamp represents the number of seconds that have elapsed since 00:00:00 on January 1, 1970. Note If the specified expiration time is larger than 30 days, the server identifies the time as the UNIX timestamp.	No	0, indicating that the values never expire
timeout	The timeout period to connect to Redis when data is written to Redis. Unit: milliseconds.	No	30,000

Parameter	Description	Required	Default value
dateFormat	The format in which the data of the DATE type is written to Redis. Set the value to yyyy-MM-dd HH:mm:ss.	No	None
writeMode	<p>The write mode. Redis supports diverse data types for values, including STRING, LIST, SET, ZSET (sorted set), and HASH. Redis Writer allows you to write values of the preceding types to Redis. The value of the writeMode parameter varies based on the specified data type of the values.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note When you configure Redis Writer, you can choose only one of the five data types described in the following table. If you do not specify a data type, the data type is STRING by default.</p> </div>	No	<i>string</i>

The following table lists the data types supported by Redis Writer.

Type	Parameter	Description	Required
String	type	The data type of the values is STRING.	Yes
	mode	The mode in which data of the STRING type is written to Redis.	Yes. Valid value: set (overwrites the existing data).
	valueFieldDelimiter	<p>This parameter is required if two or more columns are specified as the values. This parameter is not required if only one column is specified as the values.</p> <p>The delimiter used to separate values if the data is of the STRING type. Example: value1\u0001value2\u0001value3.</p>	No. Default value: \u0001.
LIST	type	The data type of the values is LIST.	Yes
	mode	The mode in which data of the LIST type is written to Redis.	Yes. Valid values: lpush (stores the data at the leftmost of the list) and rpush (stores the data at the rightmost of the list).

Type	Parameter	Description	Required
<pre> writeMode":{ "type": "list", "mode": " push rpush", "valueFieldDelimiter" : "\u0001" } </pre>	valueFieldDelimiter	The delimiter used to separate values if the data is of the STRING type. Example: value1\u0001value2\u0001value3.	No. Default value: \u0001.
	type	The data type of the values is SET.	Yes
	mode	The mode in which data of the SET type is written to Redis.	Yes. Valid value: sadd (stores the data to a set, or overwrites the existing data).
<pre> writeMode":{ "type": "set", "mode": "sadd", "valueFieldDelimiter" : "\u0001" } </pre>	valueFieldDelimiter	The delimiter used to separate values if the data is of the STRING type. Example: value1\u0001value2\u0001value3.	No. Default value: \u0001.
	type	The data type of the values is ZSET.	Yes
	mode	The mode in which data of the ZSET type is written to Redis.	Yes. Valid value: zadd (stores data to a sorted set, or overwrites the existing data).
<pre> writeMode":{ "type": "zset", "mode": "zadd" } </pre>	type	<p>The data type of the values is ZSET.</p> <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note If the data type of the values is ZSET, each data record must follow the following standard: Except for the key, a data record can contain only one score and one value. The score must be placed before the value. In this way, Redis Writer can identify which column is the score and which column is the value.</p> </div>	Yes
	mode	The mode in which data of the ZSET type is written to Redis.	Yes. Valid value: zadd (stores data to a sorted set, or overwrites the existing data).

Type	Parameter	Description	Required
HASH	type	<p>The data type of the values is HASH.</p> <div style="background-color: #e0f2f1; padding: 5px;"> <p> Note If the data type of the values is HASH, each data record must follow the following standards: Except for the key, a data record can contain only one attribute and one value. The attribute must be placed before the value. In this way, Redis Writer can identify which column is the attribute and which column is the value.</p> </div>	Yes
	mode	<p>The mode in which data of the HASH type is written to Redis.</p>	<p>Yes. Valid value: hmset (stores data to a hash sorted set, or overwrites the existing data).</p> <p>If you do not specify a data type, the data type is STRING by default.</p>

```

"writeMode":{
  "type": "hash",
  "mode": "hset"
}

```

Configure Redis Writer by using the codeless UI

The codeless UI is not supported for Redis Reader.

Configure Redis Writer by using the code editor

In the following code, a node is configured to write data to Redis. For more information about parameters, see the preceding parameter description.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "redis", // The writer type.
      "parameter": {
        "expireTime": { // The expiration time of the values to be cached in Redis.
          "seconds": "1000"
        }
      }
    }
  ]
}

```

```
    },
    "keyFieldDelimiter": "u0001", // The delimiter used to separate keys when data is written to Redis.
    "dateFormat": "yyyy-MM-dd HH:mm:ss", // The format in which the data of the DATE type is written to Redis.
    "datasource": "", // The connection name.
    "writeMode": {}, // The write mode.
    "mode": "", // The write mode used to write data of a specified data type.
    "valueFieldDelimiter": "", // The delimiter used to separate values.
    "type": "" // The data type of the values.
  },
  "keyIndexes": [ // The columns used as the key.
    0,
    1
  ],
  "batchSize": "1000" // The number of data records to write at a time.
},
"name": "Writer",
"category": "writer"
}
],
"setting": {
  "errorLimit": {
    "record": "0" // The maximum number of dirty data records allowed.
  },
  "speed": {
    "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent": 1 // The maximum number of concurrent threads.
  }
},
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}Writer"
}
]
}
}
```

2.4.5.4.17. Configure SQL Server Writer

This topic describes the data types and parameters supported by SQL Server Writer and how to configure it by using the codeless UI and code editor.

SQL Server Writer allows you to write data to tables stored in primary SQL Server databases. SQL Server Writer connects to a remote SQL Server database by using JDBC, and executes an `INSERT INTO` statement to write data to the SQL Server database. Internally, data is submitted to the database in batches.

SQL Server Writer is designed for ETL developers to import data from data warehouses to SQL Server databases. SQL Server Writer can also be used as a data migration tool by users such as DBAs.

SQL Server Writer obtains data from a Data Integration reader, and generates the `INSERT INTO` statement. If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows. To improve performance, SQL Server Writer makes batch updates with the `PreparedStatement` method and sets `rewriteBatchedStatements` to true. In this way, SQL Server Writer buffers data, and submits a write request when the amount of data in the buffer reaches a specific threshold.

 **Note**

- Data can be written only to tables stored in the primary SQL Server database.
- A sync node that uses SQL Server Writer must have at least the permission to execute the `INSERT INTO` statement. Whether other permissions are required depends on the SQL statements specified in the `preSql` and `postSql` parameters when you configure the node.

Data types

SQL Server Writer supports most SQL Server data types. Make sure that your data types are supported.

The following table lists the data types supported by SQL Server Writer.

Category	SQL Server data type
Integer	BIGINT, INT, SMALLINT, and TINYINT
Floating point	FLOAT, DECIMAL, REAL, and NUMERIC
String	CHAR, NCHAR, NTEXT, NVARCHAR, TEXT, VARCHAR, NVARCHAR (MAX), and VARCHAR (MAX)
Date and time	DATE, TIME, and DATETIME
Boolean	BIT
Binary	BINARY, VARBINARY, VARBINARY (MAX), and TIMESTAMP

Parameters

Parameter	Description	Required	Default value
<code>datasource</code>	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
<code>table</code>	The name of the destination table.	Yes	None
<code>column</code>	The columns in the destination table to which data is written. Separate the columns with commas (.). Example: <code>"column": ["id","name","age"]</code> . To write data to all the columns in the destination table, set the value to an asterisk (*), for example, <code>"column":["*"]</code> .	Yes	None
<code>preSql</code>	The SQL statement to execute before the sync node is run. For example, you can clear outdated data before data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None

Parameter	Description	Required	Default value
postSql	The SQL statement to execute after the sync node is run. For example, you can add a timestamp after data synchronization. You can execute only one SQL statement on the codeless UI and multiple SQL statements in the code editor.	No	None
writeMode	The write mode. Valid value: <i>insert</i> . When a data record violates the primary key constraint or unique index constraint, Data Integration considers it dirty and retains the original data.	No	<i>insert</i>
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the SQL Server database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,024

Configure SQL Serve Writer by using the codeless UI

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Statement Run Before Writing	The preSql parameter in the preceding parameter description. Enter an SQL statement to execute before the sync node is run.
Statement Run After Writing	The postSql parameter in the preceding parameter description. Enter an SQL statement to execute after the sync node is run.
Solution to Primary Key Violation	The writeMode parameter in the preceding parameter description. Select the required write mode.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description.

Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.

GUI element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.

GUI element	Description
Auto Layout	Click Auto Layout to sort the fields based on specified rules.
Change Fields	Click the Change Fields icon. In the Change Fields dialog box, you can manually edit fields in the source table. Each field occupies a row. The first and the last blank rows are included, whereas other blank rows are ignored.
Add	<ul style="list-style-type: none"> ◦ Click Add to add a field. You can enter constants. Each constant must be enclosed in single quotation marks (' '), for example, 'abc' and '123'. ◦ You can use scheduling parameters such as \${bizdate}. ◦ You can enter functions supported by relational databases, for example, now() and count(1). ◦ If the value you entered cannot be parsed, the type is displayed as Unidentified.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure SQL Server Writer by using the code editor

In the following code, a node is configured to write data to an SQL Server database. For more information about the parameters, see the preceding parameter description.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"sqlserver",// The writer type.
      "parameter":{
        "postSql":[// The SQL statement to execute after the sync node is run.
          "datasource":"","// The connection name.
          "column":[// The columns to which data is written.
            "id",
            "name"
          ],
          "table":"","// The name of the destination table.
          "preSql":[// The SQL statement to execute before the sync node is run.
        ],
        "name":"Writer",
        "category":"writer"
      }
    },
    ],
    "setting":{
      "errorLimit":{
        "record":"0"// The maximum number of dirty data records allowed.
      },
      "speed":{
        "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
        "concurrent":1 // The maximum number of concurrent threads.
      }
    },
    "order":{
      "hops":[
        {
          "from":"Reader",
          "to":"Writer"
        }
      ]
    }
  }
}
```

2.4.5.4.18. Configure Elasticsearch Writer

This topic describes the data types and parameters supported by Elasticsearch Writer and how to configure it by using the code editor.

Elasticsearch is an open-source product that complies with the Apache open standards. It is the mainstream search engine for enterprise data. Elasticsearch is a Lucene-based data search and analysis tool that provides distributed services. The mappings between Elasticsearch core concepts and database core concepts are as follows:

Relational database (instance) -> database -> table -> row -> column
 Elasticsearch -> index -> type -> document -> field

Elasticsearch can contain multiple indexes (databases). Each index can contain multiple types (tables). Each type can contain multiple documents (rows). Each document can contain multiple fields (columns). Elasticsearch Writer uses the RESTful API of Elasticsearch to write multiple data records retrieved by a reader to Elasticsearch at a time.

Parameters

Parameter	Description	Required	Default value
endpoint	The endpoint for accessing Elasticsearch, in the format of <code>http://xxx.com:9999</code> .	No	None
accessId	The AccessKey ID for accessing Elasticsearch, which is used for authorization when a connection with Elasticsearch is established. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note The accessId and accessKey parameters are required. If you do not set the parameters, an error is returned. If you use on-premises Elasticsearch for which basic authentication is not configured, the AccessKey ID and AccessKey secret are not required. In this case, you can set the accessId and accessKey parameters to random values.</p> </div>	No	None
accessKey	The AccessKey secret for accessing Elasticsearch.	No	None
index	The index name in Elasticsearch.	No	None
indexType	The type name in the index of Elasticsearch.	No	<i>Elasticsearch</i>
cleanup	Specifies whether to clear existing data in the index. The method used to clear the data is to delete and rebuild the corresponding index. The default value false indicates that the existing data in the index is retained.	No	<i>false</i>
batchSize	The number of data records to write at a time.	No	<i>1000</i>
trySize	The number of retries after a failure.	No	<i>30</i>
timeout	The connection timeout of the client. Unit: milliseconds.	No	<i>600000</i>
discovery	Specifies whether to enable Node Discovery. When Node Discovery is enabled, the server list in the client is polled and regularly updated.	No	<i>false</i>

Parameter	Description	Required	Default value
compression	Specifies whether to enable compression for an HTTP request.	No	<i>true</i>
multiThread	Specifies whether to use multiple threads for an HTTP request.	No	<i>true</i>
ignoreWriteError	Specifies whether to ignore write errors and proceed with writing without retries.	No	<i>false</i>
ignoreParseError	Specifies whether to ignore format parsing errors and proceed with writing.	No	<i>true</i>
alias	<p>The alias of the index. The alias feature of Elasticsearch is similar to the view feature of a traditional database. For example, if you create an alias named <code>my_index_alias</code> for the index <code>my_index</code>, the operations on <code>my_index_alias</code> also take effect on <code>my_index</code>.</p> <p>Configuring alias means that after the data import is completed, an alias is created for the specified index.</p>	No	None
aliasMode	The mode in which an alias is added after the data is imported. Valid values: <i>append</i> and <i>exclusive</i> .	No	append
settings	<p>The delimiter (-,-) for splitting the source data if you are inserting an array to Elasticsearch. Example:</p> <p>The source column stores data <code>a,-,b,-,c,-,d</code> of the String type. Elasticsearch Writer uses the delimiter (-,-) to split the source data and obtains the array <code>["a", "b", "c", "d"]</code>. Then, Elasticsearch Writer writes the array to the corresponding field in Elasticsearch.</p>	No	-,-
	<p>The fields of the document. The parameters for each field include basic parameters such as name and type and advanced parameters such as analyzer, format, and array.</p> <p>The field types supported by Elasticsearch are as follows:</p>		

Parameter	Description	Required	Default value
column	<p>- id // The id type corresponds to the _id type in Elasticsearch, and can be considered as the unique primary key. Data with the same ID will be overwritten and not indexed.</p> <ul style="list-style-type: none"> - string - text - keyword - long - integer - short - byte - double - float - date - boolean - binary - integer_range - float_range - long_range - double_range - date_range - geo_point - geo_shape - ip - token_count - array - object - nested <ul style="list-style-type: none"> When the field type is Text, you can specify the analyzer, norms, and index_options parameters. Example: <pre data-bbox="475 1234 1050 1413" style="background-color: #f0f0f0; padding: 5px;"> { "name": "col_text", "type": "text", "analyzer": "ik_max_word" } </pre> When the field type is date, you can specify the format and timezone parameters, indicating the date serialization format and the time zone, respectively. Example: <pre data-bbox="475 1559 1050 1765" style="background-color: #f0f0f0; padding: 5px;"> { "name": "col_date", "type": "date", "format": "yyyy-MM-dd HH:mm:ss", "timezone": "UTC" } </pre> When the field type is ge_shape, you can specify the tree (geohash or quadtree) and precision parameters. Example: 	Yes	None

Parameter	Description	Required	Default value
	<pre>{ "name": "col_geo_shape", "type": "geo_shape", "tree": "quadtree", "precision": "10m" }</pre> <p>If you specify the array parameter for a field and set the array parameter to <i>true</i>, the field is an array column. Elasticsearch Writer uses the delimiter specified by the splitter to split the source data, converts the data to an array of strings, and writes the array to the destination. Only one delimiter is supported for one node. Example:</p> <pre>{ "name": "col_integer_array", "type": "integer", "array": true }</pre>		
dynamic	Specifies whether to use the mapping configuration of Elasticsearch. A value of <i>true</i> indicates that the mapping configuration of Elasticsearch, instead of the mapping configuration of Data Integration, is used.	No	<i>false</i>
actionType	<p>The type of the action for writing data to Elasticsearch. Currently, Data Integration supports only the following action types: <i>index</i> and <i>update</i>. Default value: <i>index</i>.</p> <ul style="list-style-type: none"> • <i>index</i>: Data Integration uses Index.Builder of the Elasticsearch SDK to construct a request for writing multiple data records at a time. In <i>index</i> mode, Elasticsearch first checks whether an ID is specified for the document to be inserted. <ul style="list-style-type: none"> ◦ If the ID is not specified, Elasticsearch generates a unique ID by default. In this case, the document is directly inserted to Elasticsearch. ◦ If the ID is specified, Elasticsearch replaces the existing document with the document to be inserted. <div style="border: 1px solid #add8e6; padding: 5px; margin: 10px 0;"> <p> Note In this case, you cannot modify specific fields in the document.</p> </div> • <i>update</i>: Data Integration uses Update.Builder of the Elasticsearch SDK to construct a request for writing multiple data records at a time. In <i>update</i> mode, Elasticsearch calls the get method of InternalEngine to obtain the information of the original document for each update. In this way, you can modify specific fields. In update mode, you must obtain the information of the original document for each update, which greatly affects the performance. However, you can modify specific fields in this mode. If the original document does not exist, the new document is directly inserted. 	No	<i>index</i>

Configure Elasticsearch Writer by using the code editor

In the following code, a node is configured to write data to Elasticsearch. For more information about the parameters, see the preceding parameter description.

```
{
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  },
  "setting": {
    "errorLimit": {
      "record": "0"
    },
    "speed": {
      "concurrent": 1,
      "throttle": false
    }
  },
  "steps": [
    {
      "category": "reader",
      "name": "Reader",
      "parameter": {
      },
      "stepType": "stream"
    },
    {
      "category": "writer",
      "name": "Writer",
      "parameter": {
        "endpoint": "http://xxx.com:9999",
        "accessId": "xxx",
        "accessKey": "yyy",
        "index": "test-1",
        "type": "default",
        "cleanup": true,
        "settings": {
          "index": {
            "number_of_shards": 1,
            "number_of_replicas": 0
          }
        }
      },
      "discovery": false,
      "batchSize": 1000,
      "splitter": ",",
      "column": [
        {
          "name": "pk",
          "type": "id"
        },
        {
          "name": "col_ip",
          "type": "ip"
        }
      ]
    }
  ]
}
```

```
    },
    {
      "name": "col_double",
      "type": "double"
    },
    {
      "name": "col_long",
      "type": "long"
    },
    {
      "name": "col_integer",
      "type": "integer"
    },
    {
      "name": "col_keyword",
      "type": "keyword"
    },
    {
      "name": "col_text",
      "type": "text",
      "analyzer": "ik_max_word"
    },
    {
      "name": "col_geo_point",
      "type": "geo_point"
    },
    {
      "name": "col_date",
      "type": "date",
      "format": "yyyy-MM-dd HH:mm:ss"
    },
    {
      "name": "col_nested1",
      "type": "nested"
    },
    {
      "name": "col_nested2",
      "type": "nested"
    },
    {
      "name": "col_object1",
      "type": "object"
    },
    {
      "name": "col_object2",
      "type": "object"
    },
    {
      "name": "col_integer_array",
      "type": "integer",
      "array": true
    },
    {
      "name": "col_geo_shape",
      "type": "geo_shape",
      "tree": "quadtree",
      "precision": "10m"
    }
  ]
}
```

```

    }
  ]
},
"stepType": "elasticsearch"
}
],
"type": "job",
"version": "2.0"
}

```

Note Currently, Elasticsearch that is deployed in a Virtual Private Cloud (VPC) supports only custom resource groups. A sync node that is run on the default resource group may fail to connect to Elasticsearch.

2.4.5.4.19. Configure LogHub Writer

This topic describes the data types and parameters supported by LogHub Writer and how to configure it by using the code editor.

LogHub Writer allows you to transfer data from a Data Integration reader to LogHub through Log Service Java SDK.

Note LogHub does not guarantee idempotence. Rerunning a node after the node fails may result in redundant data.

LogHub Writer obtains data from a Data Integration reader and converts the data types supported by Data Integration to String. When the number of the data records reaches the value specified for the batchSize parameter, LogHub Writer sends the data records to LogHub at a time through Log Service Java SDK. LogHub Writer sends 1,024 data records at a time by default. The batchSize parameter can be set to 4096 at most.

Data types

The following table lists the data types supported by LogHub Writer.

Data Integration data type	LogHub data type
LONG	STRING
DOUBLE	STRING
STRING	STRING
DATE	STRING
BOOLEAN	STRING
BYTES	STRING

Parameters

Parameter	Description	Required	Default value
endpoint	The endpoint for accessing Log Service.	Yes	None

Parameter	Description	Required	Default value
accessKeyId	The AccessKey ID for accessing Log Service.	Yes	None
accessKeySecret	The AccessKey secret for accessing Log Service.	Yes	None
project	The name of the destination Log Service project.	Yes	None
logstore	The name of the destination Logstore.	Yes	None
topic	The name of the destination topic.	No	Empty string
batchSize	The number of data records to write at a time.	No	1024
column	The columns in each data record.	Yes	None

Configure LogHub Writer by using the codeless UI

Currently, the codeless user interface (UI) is not supported for LogHub Writer.

Configure LogHub Writer by using the code editor

In the following code, a node is configured to write data to LogHub. For more information about the parameters, see the preceding parameter description.

```

{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    { //
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "loghub", // The writer type.
      "parameter": {
        "datasource": "", // The connection name.
        "column": [ // The columns in each data record.
          "col0",
          "col1",
          "col2",
          "col3",
          "col4",
          "col5"
        ],
        "topic": "", // The name of the destination topic.
        "batchSize": "1024", // The number of data records to write at a time.
        "logstore": "" // The name of the destination Logstore.
      },
      "name": "Writer",
      "category": "writer"
    }
  ],
  "setting": {
    "errorLimit": {
      "record": "" // The maximum number of dirty data records allowed.
    },
    "speed": {
      "concurrent": 3, // The maximum number of concurrent threads.
      "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    }
  },
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  }
}

```

2.4.5.4.20. Configure Open Search Writer

This topic describes the data types and parameters supported by Open Search Writer and how to configure it by using the code editor.

How it works

Open Search Writer allows you to insert data to or update data in Open Search. Open Search Writer is designed for developers to import data to Open Search so that the data can be searched.

Specifically, Open Search Writer uses the search API provided by Open Search to import data.

Note

- Open Search V3 uses internal dependent databases, with POM of `com.aliyun.opensearch:aliyun-sdk-opensearch:2.1.3`.
- To use Open Search Writer, you must install JDK 1.6-32 or later. You can run the `java-version` command to view the JDK version.

Features

The columns in Open Search are unordered. Open Search Writer writes data strictly in accordance with the order of the specified columns. If the number of specified columns is less than that in Open Search, redundant columns in Open Search are set to the default value or null.

Assume that an Open Search table contains columns a, b, and c, and you only need to write data to columns b and c. You can set the column parameter to ["c","b"]. In this case, Open Search Writer imports the first and second columns of the source data obtained from a reader to columns c and b in the Open Search table respectively. Column a in the Open Search table is set to the default value or null.

Additional instructions:

- Handling of column configuration errors

To avoid losing the data of redundant columns and ensure high data reliability, Open Search Writer returns an error message if the number of columns to be written is more than that in the destination Open Search table. For example, if an Open Search table contains columns a, b, and c, Open Search Writer returns an error if more than three columns are to be written to the table.

- Table configuration

Open Search Writer can write data to only one table at a time.

- Node rerunning

After a node is rerun, data is automatically overwritten based on IDs. Therefore, the data written to Open Search must contain one ID column. An ID is a unique identifier of a row in Open Search. The existing data with the same ID as the new data will be overwritten.

- Node rerunning

After a node is rerun, data is automatically overwritten based on IDs.

Data types

Open Search Writer supports most Open Search data types. Make sure that your data types are supported.

The following table lists the data types supported by Open Search Writer.

Category	Open Search data type
Integer	INT
Floating point	DOUBLE and FLOAT
String	TEXT, LITERAL, and SHORT_TEXT

Category	Open Search data type
Date and time	INT
Boolean	LITERAL

Parameters

Parameter	Description	Required	Default value
accessId	The AccessKey ID for connecting to the Open Search database.	Yes	None
accessKey	The AccessKey secret for connecting to the Open Search database.	Yes	None
host	The endpoint for connecting to Open Search. You can view the endpoint in the Apsara Stack console.	Yes	None
indexName	The name of the Open Search project.	Yes	None
table	The name of the table to which data is written. You can specify only one table name because Data Integration does not support importing data to multiple tables at a time.	Yes	None
column	<p>The columns in the destination table to which data is written. To write data to all the columns in the destination table, set the value to an asterisk (*), for example, <code>"column":["*"]</code>. Separate the columns with a comma (,) if data is written to some of the columns in the destination table. Example: <code>"column":["id","name"]</code>.</p> <p>Open Search Writer supports filtering columns and changing the order of columns. Assume that an Open Search table has three columns: a, b, and c. If you want to write data only to columns c and b, you can set the column parameter in the format <code>"column":["c","b"]</code>. During data synchronization, column a is automatically set to null.</p>	Yes	None
batchSize	<p>The number of data records to write at a time. Data is written to Open Search in batches. The advantage of Open Search is data query. The transactions per second (TPS) of Open Search is generally not high. Set this parameter based on the resources available for the account that is used to connect to Open Search.</p> <p>Generally, the size of a data record must be less than 1 MB, and the size of the data records to write at a time must be less than 2 MB.</p>	Required only for writing data to a partitioned table	300

Parameter	Description	Required	Default value
writeMode	<p>The write mode. To ensure the idempotence of write operations, set the writeMode parameter to add/update when you configure Open Search Writer.</p> <ul style="list-style-type: none"> • add: deletes the existing data record and inserts the new data record to Open Search, which is an atomic operation. • update: updates the existing data record based on the new data record, which is an atomic operation. <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note Writing data to Open Search in batches is not an atomic operation. Part of the data may fail to be written. Exercise caution when you set the writeMode parameter. Open Search V3 does not support the update mode.</p> </div>	Yes	None
ignoreWriteError	<p>Specifies whether to ignore failed write operations.</p> <p>Example: <code>"ignoreWriteError":true</code> . If data is written to Open Search in batches, this parameter specifies whether to ignore failed write operations in the current batch. If you set the parameter to true, Open Search Writer continues to perform other write operations. If you set the parameter to false, the sync node ends and an error message is returned. We recommend that you use the default value.</p>	No	<i>false</i>
version	<p>The version of Open Search, for example, <code>"version":"v3"</code> . We recommend that you use Open Search V3 because the push operation faces many constraints in Open Search V2.</p>	No	v2

Configure Open Search Writer by using the code editor

In the following code, a node is configured to write data to Open Search.

```

{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "reader": {},
    "writer": {
      "plugin": "opensearch",
      "parameter": {
        "accessId": "*****",
        "accessKey": "*****",
        "host": "http://yyyy.aliyuncs.com",
        "indexName": "datax_xxx",
        "table": "datax_yyy",
        "column": [
          "appkey",
          "id",
          "title",
          "gmt_create",
          "pic_default"
        ],
        "batchSize": 500,
        "writeMode": add,
        "version": "v2",
        "ignoreWriteError": false
      }
    }
  }
}

```

2.4.5.4.21. Configure Tablestore Writer

This topic describes the data types and parameters supported by Tablestore Writer and how to configure it by using the code editor.

Tablestore is a NoSQL database service that is built on the Apsara distributed operating system. The service allows you to store and access large volumes of structured data in real time. Tablestore organizes data into instances and tables. It uses data sharding and load balancing technologies to seamlessly expand the data scale.

Tablestore Writer connects to the Tablestore server by using Tablestore SDK for Java and writes data to the server by using the SDK. Tablestore Writer greatly optimizes the write process, including retry after write timeouts, retry after exceptions, and batch submission.

Tablestore Writer writes data to Tablestore in one of the following modes:

- **PutRow**: the PutRow API operation for Tablestore, which is used to insert data to a specified row. If this row does not exist, a new row is added. Otherwise, the original row is overwritten.
- **UpdateRow**: the UpdateRow API operation for Tablestore, which is used to update the data of a specified row. If this row does not exist, a new row is added. Otherwise, the values of the specified columns are added, modified, or removed as requested.

Tablestore Writer supports all Tablestore data types. The following table lists the data types.

Category	Tablestore data type
Integer	INTEGER

Category	Tablestore data type
Floating point	DOUBLE
String	STRING
Boolean	BOOLEAN
Binary	BINARY

 **Note** To write INTEGER-type data, set the data type to INT in the code editor. Then, DataWorks converts the INT type into the INTEGER type. If you directly set the data type to INTEGER, an error is reported in the log, and the node fails.

Parameters

Parameter	Description	Required	Default value
datasource	The name of the data source. It must be the same as the name of the added data source. You can add data sources by using the code editor.	Yes	None
endPoint	The endpoint of the Tablestore server.	Yes	None
accessId	The AccessKey ID of the account that is used to access Tablestore.	Yes	None
accessKey	The AccessKey secret of the account that is used to access Tablestore.	Yes	None
instanceName	The name of the Tablestore instance. The instance is an entity for you to use and manage Tablestore. After you activate the Tablestore service, you must create an instance in the Tablestore console before you can create and manage tables. Instances are the basic units that you can use to manage Tablestore resources. Access control and resource measurement for applications are implemented at the instance level.	Yes	None
table	The name of the destination table. You can specify only one table as the destination table. Multi-table synchronization is not required for Tablestore.	Yes	None

Parameter	Description	Required	Default value
primaryKey	<p>The primary key of the destination table in Tablestore. The primary keys are described in a JSON array. Tablestore is a NoSQL database service. You must specify the primary key of the destination table for Tablestore Writer to write data.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note The primary keys in Tablestore must be of the STRING or INT type. Therefore, you must set the data type of a primary key to STRING or INT in the code editor.</p> </div> <p>Data Integration supports data type conversion. Tablestore Writer can convert data that is not of the STRING or INT type to the STRING or INT type. The following code provides a configuration example:</p> <pre style="background-color: #f0f0f0; padding: 5px;">"primaryKey": [{"name": "pk1", "type": "string"}, {"name": "pk2", "type": "int"}],</pre>	Yes	None
column	<p>The columns that you want to synchronize to the destination table. The columns are described in a JSON array.</p> <p>Specify this parameter in the following format:</p> <pre style="background-color: #f0f0f0; padding: 5px;">{"name": "col2", "type": "INT"},</pre> <p>The name parameter specifies the name of the column to which data is written. The type parameter specifies the data type of the column. Data types supported by Tablestore include STRING, INT, DOUBLE, BOOLEAN, and BINARY.</p>	Yes	None
writeMode	<p>The write mode. Constants, functions, or custom statements are not supported during the write process. The following three modes are supported:</p> <ul style="list-style-type: none"> PutRow: the PutRow API operation for Tablestore, which is used to insert data to a specified row. If this row does not exist, a new row is added. Otherwise, the original row is overwritten. UpdateRow: the UpdateRow API operation for Tablestore, which is used to update the data of a specified row. If this row does not exist, a new row is added. Otherwise, the values of the specified columns are added, modified, or removed as requested. DeleteRow: deletes a row. 	Yes	None

Configure Tablestore Writer by using the codeless UI

This method is not supported.

Configure Tablestore Writer by using the code editor

In the following code, a node is configured to write data to Tablestore:

```
{
```

```
"type":"job",
"version":"2.0",// The version number.
"steps":[
  {
    "stepType":"stream",
    "parameter":{},
    "name":"Reader",
    "category":"reader"
  },
  {
    "stepType":"ots",// The writer type.
    "parameter":{
      "datasource":"","// The data source.
      "column":[// The columns to which data is written.
        {
          "name":"columnName1",// The name of the column.
          "type":"INT",// The data type of the column.
        },
        {
          "name":"columnName2",
          "type":"STRING"
        },
        {
          "name":"columnName3",
          "type":"DOUBLE"
        },
        {
          "name":"columnName4",
          "type":"BOOLEAN"
        },
        {
          "name":"columnName5",
          "type":"BINARY"
        }
      ],
      "writeMode":"","// The write mode.
      "table":"","// The name of the destination table.
      "primaryKey":[// The primary key of the destination table in Tablestore.
        {
          "name":"pk1",
          "type":"STRING"
        },
        {
          "name":"pk2",
          "type":"INT"
        }
      ]
    },
    "name":"Writer",
    "category":"writer"
  }
],
"setting":{
  "errorLimit":{
    "record":"0",// The maximum number of dirty data records allowed.
  },
  "speed":{
```

```
"throttle":false,// Specifies whether to enable bandwidth throttling. The value false indicates that bandwidth throttling is disabled, and the value true indicates that bandwidth throttling is enabled. The concurrent parameter takes effect only when the throttle parameter is set to true.
```

```
    "concurrent":1,// The maximum number of concurrent threads.
  }
},
"order":{
  "hops":[
    {
      "from":"Reader",
      "to":"Writer"
    }
  ]
}
}
```

2.4.5.4.22. Configure RDBMS Writer

This topic describes the data types and parameters supported by RDBMS Writer and how to configure it by using the code editor.

RDBMS Writer allows you to write data to tables stored in primary relational database management system (RDBMS) databases. Specifically, RDBMS Writer obtains data from a Data Integration reader, connects to a remote RDBMS database through Java Database Connectivity (JDBC), and then runs an `INSERT INTO` statement to write data to the RDBMS database. RDBMS Writer is a common writer for relational databases. To enable RDBMS Writer to support a new relational database, register the driver for the relational database.

RDBMS Writer is designed for extract-transform-load (ETL) developers to import data from data warehouses to RDBMS databases. RDBMS Writer can also be used as a data migration tool by users such as database administrators (DBAs).

Data types

RDBMS Writer supports most data types in relational databases, such as numbers and characters. Make sure that your data types are supported.

Parameters

Parameter	Description	Required	Default value
jdbcUrl	<p>The JDBC URL for connecting to the database. The format must be in accordance with official specifications. You can also specify the information of the attachment facility. The format varies with the database type. Data Integration selects an appropriate driver for data reading based on the format.</p> <ul style="list-style-type: none"> Format for DM databases: <code>jdbc:dm://ip:port/database</code> Format for Db2 databases: <code>jdbc:db2://ip:port/database</code> Format for PPAS databases: <code>jdbc:edb://ip:port/database</code> 	Yes	None
username	The username for connecting to the database.	Yes	None

Parameter	Description	Required	Default value
password	The password for connecting to the database.	Yes	None
table	The name of the destination table.	Yes	None
column	<p>The columns in the destination table to which data is written. Separate the columns with a comma (,).</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>? Note We recommend that you do not use the default setting.</p> </div>	Yes	None
preSql	<p>The SQL statement to run before the sync node is run. For example, you can clear outdated data before data synchronization. Currently, you can run only one SQL statement.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>? Note If you specify multiple SQL statements in the code editor, the system does not guarantee that they are run in the same transaction.</p> </div>	No	None
postSql	<p>The SQL statement to run after the sync node is run. For example, you can add a timestamp after data synchronization. Currently, you can run only one SQL statement.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>? Note If you specify multiple SQL statements in the code editor, the system does not guarantee that they are run in the same transaction.</p> </div>	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the RDBMS database over the network, and increase the throughput. However, an excessively large value may lead to the out of memory (OOM) error during the data synchronization process.	No	1024

Configure RDBMS Writer by using the code editor

In the following code, a node is configured to write data to an RDBMS database.

```
{
  "job": {
    "setting": {
      "speed": {
        "channel": 1
      }
    }
  },
  "content": [
    {
      "reader": {
```

```

    "name": "streamreader",
    "parameter": {
      "column": [
        {
          "value": "DataX",
          "type": "string"
        },
        {
          "value": 19880808,
          "type": "long"
        },
        {
          "value": "1988-08-08 08:08:08",
          "type": "date"
        },
        {
          "value": true,
          "type": "bool"
        },
        {
          "value": "test",
          "type": "bytes"
        }
      ],
      "sliceRecordCount": 1000
    }
  },
  "writer": {
    "name": "RDBMS Writer",
    "parameter": {
      "connection": [
        {
          "jdbcUrl": "jdbc:dm://ip:port/database",
          "table": [
            "table"
          ]
        }
      ],
      "username": "username",
      "password": "password",
      "table": "table",
      "column": [
        "*"
      ],
      "preSql": [
        "delete from XXX;"
      ]
    }
  }
}
]
}
}

```

You can enable RDBMS Writer to support a new database as follows:

1. Go to the directory of RDBMS Writer, `#{DATA_HOME}/plugin/writer/RDBMS Writer`. In the preceding

directory, `$(DATA_HOME)` indicates the main directory of Data Integration.

2. Add the driver of your database to the `drivers` array in the `plugin.json` file in the RDBMS Writer directory. RDBMS Writer automatically selects an appropriate driver for connecting to a database.

```
{
  "name": "RDBMS Writer",
  "class": "com.alibaba.datax.plugin.reader.RDBMS Writer.RDBMS Writer",
  "description": "useScene: prod. mechanism: Jdbc connection using the database, execute select sql, retrieve data from the ResultSet. warn: The more you know about the database, the less problems you encounter.",
  "developer": "alibaba",
  "drivers": [
    "dm.jdbc.driver.DmDriver",
    "com.ibm.db2.jcc.DB2Driver",
    "com.sybase.jdbc3.jdbc.SybDriver",
    "com.edb.Driver"
  ]
}
```

3. Add the package of the driver to the `libs` directory in the RDBMS Writer directory.

```
$tree
.
|-- libs
| |-- Dm7JdbcDriver16.jar
| |-- commons-collections-3.0.jar
| |-- commons-io-2.4.jar
| |-- commons-lang3-3.3.2.jar
| |-- commons-math3-3.1.1.jar
| |-- datax-common-0.0.1-SNAPSHOT.jar
| |-- datax-service-face-1.0.23-20160120.024328-1.jar
| |-- db2jcc4.jar
| |-- druid-1.0.15.jar
| |-- edb-jdbc16.jar
| |-- fastjson-1.1.46.sec01.jar
| |-- guava-r05.jar
| |-- hamcrest-core-1.3.jar
| |-- jconn3-1.0.0-SNAPSHOT.jar
| |-- logback-classic-1.0.13.jar
| |-- logback-core-1.0.13.jar
| |-- plugin-rdbms-util-0.0.1-SNAPSHOT.jar
| `-- slf4j-api-1.7.10.jar
|-- plugin.json
|-- plugin_job_template.json
`-- RDBMS Writer-0.0.1-SNAPSHOT.jar
```

2.4.5.4.23. Configure Stream Writer

This topic describes the data types and parameters supported by Stream Writer and how to configure it by using the code editor.

Stream Writer allows you to display the data obtained from a Data Integration reader on the screen or discard the data. Stream Writer is mainly applicable to performance testing for data synchronization and basic functional testing.

Parameters

print

- Description: specifies whether to display the data obtained from the reader on the screen.
- Required: No
- Default value: true

Configure Stream Writer by using the codeless UI

Currently, the codeless user interface (UI) is not supported for Stream Writer.

Configure Stream Writer by using the code editor

In the following code, a node is configured to display the data obtained from a Data Integration reader on the screen.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"stream",// The writer type.
      "parameter":{
        "print": false, // Specifies whether to display data on the screen.
        "fieldDelimiter":",",// The column delimiter.
      },
      "name":"Writer",
      "category":"writer"
    }
  ],
  "setting":{
    "errorLimit":{
      "record":"0",// The maximum number of dirty data records allowed.
    },
    "speed":{
      "throttle":false,// Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
      "concurrent":1,// The maximum number of concurrent threads.
    }
  },
  "order":{
    "hops":[
      {
        "from":"Reader",
        "to":"Writer"
      }
    ]
  }
}
```

2.4.5.4.24. Configure Hive Writer

Hive Writer allows you to write data to HDFS and load the data to Hive. This topic describes how Hive Writer works, its parameters, and how to configure it by using the codeless UI and code editor.

Background information

Hive is a Hadoop-based data warehouse tool that is used to process large amounts of structured logs. Hive maps structured data files to a table and allows you to execute SQL statements to query data in the table.

Essentially, Hive converts Hive Query Language (HQL) or SQL statements to MapReduce programs.

- Hive stores processed data in HDFS.
- Hive uses MapReduce programs to analyze data at the underlying layer.
- Hive runs MapReduce programs on Yarn.

How it works

Hive Writer accesses a Hive metastore, parses the configuration to obtain the file storage path, file format, and column delimiter of the file to which data is written, and then writes data to the HDFS file. Hive Writer loads data in the HDFS file to the destination Hive table by using JDBC.

The underlying logic of Hive Writer is the same as that of HDFS Writer. You can configure parameters of HDFS Writer in the parameters of Hive Writer. Data Integration transparently transmits the configured parameters to HDFS Writer.

Parameters

Parameter	Description	Required	Default value
datasource	The connection name. It must be the same as the name of the added connection.	Yes	None
column	<p>The columns to which data is written. Example: <code>"column": ["id", "name"]</code>.</p> <ul style="list-style-type: none"> • Column pruning is supported. You can select specific columns to export. • The column parameter must explicitly specify a set of columns to which data is written. The parameter cannot be left empty. • The column order cannot be changed. 	Yes	None
table	<p>The name of the Hive table to which data is written.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> ? Note The name is case-sensitive. </div>	Yes	None
partition	<p>The partition in the Hive table to which data is written.</p> <ul style="list-style-type: none"> • This parameter is required for a partitioned Hive table. The sync node writes data to the partition specified by the partition parameter. • This parameter is not required for a non-partitioned table. 	No	None

Parameter	Description	Required	Default value
writeMode	<p>The mode in which data is loaded to the Hive table. After data is written to the HDFS file, Hive Writer executes the <code>LOAD DATA INPATH (overwrite) INTO TABLE</code> statement to load data to the Hive table.</p> <p>The writeMode parameter specifies the data loading mode.</p> <ul style="list-style-type: none"> • <i>truncate</i>: deletes existing data before loading the data to the Hive table. • <i>append</i>: retains the existing data and appends the data to the Hive table. • If the writeMode parameter is set to Other, the data is written to the HDFS file but not loaded to the Hive table. <div style="background-color: #e1f5fe; padding: 10px; margin-top: 10px;"> <p> Note Setting the writeMode parameter is a high-risk operation. Pay attention to the destination directory and the value of this parameter to avoid deleting data incorrectly.</p> <p>This parameter must be used together with the hiveConfig parameter.</p> </div>	Yes	None

Parameter	Description	Required	Default value
hiveConfig	<p>The extended parameters for Hive, including hiveCommand, jdbcUrl, username, and password.</p> <ul style="list-style-type: none"> hiveCommand: the full path of the Hive client. After you run the <code>hive -e</code> command, the <code>LOAD DATA INPATH</code> statement is executed to load data based on the mode specified by the writeMode parameter. <p>The client specified by the hiveCommand parameter provides access information about Hive.</p> <ul style="list-style-type: none"> jdbcUrl, username, and password: the information that is required to connect to Hive by using JDBC. After Hive Writer connects to Hive by using JDBC, Hive Writer executes the <code>LOAD DATA INPATH</code> statement to load data based on the mode specified by the writeMode parameter. <pre>"hiveConfig": { "hiveCommand": "", "jdbcUrl": "", "username": "", "password": "" }</pre> <ul style="list-style-type: none"> Hive Writer allows you to write data to HDFS files by using an HDFS client. You can use the hiveConfig parameter to specify advanced settings for the HDFS client. 	Yes	None

Configure Hive Writer by using the codeless UI

On the DataStudio page, double-click a data sync node, and perform the following operations on the node configuration tab that appears:

1. Configure the connections.

Configure the connections to the source and destination data stores for the sync node.

Parameter	Description
Connection	The datasource parameter in the preceding parameter description. Select a connection type, and then select a connection name that you have configured in DataWorks.
Table	The table parameter in the preceding parameter description.
Partition Key Column	The partition to which data is written. The last-level partition must be specified. Hive Writer can write data to only one partition.
Writing Rule	The writeMode parameter in the preceding parameter description.

2. Configure field mapping. It is equivalent to setting the column parameter in the preceding parameter description. Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right.

GUI Element	Description
Map Fields with the Same Name	Click Map Fields with the Same Name to establish a mapping between fields with the same name. Note that the data types of the fields must match.
Map Fields in the Same Line	Click Map Fields in the Same Line to establish a mapping for fields in the same row. Note that the data types of the fields must match.
Delete All Mappings	Click Delete All Mappings to remove mappings that have been established.
Auto Layout	Click Auto Layout to sort the fields based on specified rules.

3. Configure channel control policies.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from and write data to data storage within the sync node. You can configure the concurrency for a node on the codeless UI.
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.
Resource Group	The resource group used for running the sync node. By default, the node runs on the default resource group. If resources are insufficient, you can add a custom resource group and run the sync node on the custom resource group. Set the resource group properly based on network conditions of the connections, resource group usage, and business importance.

Configure Hive Writer by using the code editor

In the following code, a node is configured to write data to Hive in JSON format.

```
{
  "type": "job",
  "steps": [
    {
      "stepType": "hive",
      "parameter": {
      },
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "hive",
      "parameter": {
        "partition": "year=a,month=b,day=c", // The partition to which data is written.
        "datasource": "hive_ha_shanghai", // The connection name.
        "table": "partitiontable2", // The name of the destination table.
        "column": [ // The columns in the destination table to which data is written.
          "id",
          "name",
          "age"
        ],
        "writeMode": "append" // The write mode.
      },
      "name": "Writer",
      "category": "writer"
    }
  ],
  "version": "2.0",
  "order": {
    "hops": [
      {
        "from": "Reader",
        "to": "Writer"
      }
    ]
  },
  "setting": {
    "errorLimit": {
      "record": ""
    },
    "speed": {
      "throttle": false,
      "concurrent": 2
    }
  }
}
```

2.4.5.4.25. Configure Vertica Writer

Vertica is a column-oriented database using the MPP architecture. Vertica Writer allows you to write data to tables stored in Vertica databases. This topic describes how Vertica Writer works, its parameters, and how to configure it by using the code editor.

How it works

Vertica Writer connects to a remote Vertica database by using JDBC, and executes an `INSERT INTO` statement to write data to the Vertica database. Internally, data is submitted to the Vertica database in batches.

Vertica Writer is designed for ETL developers to import data from data warehouses to Vertica databases. Vertica Writer can also be used as a data migration tool by users such as DBAs.

Vertica Writer obtains data from a Data Integration reader, and generates the `INSERT INTO` statement based on your configurations.

- `INSERT INTO` : If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows.
- Data can be written only to tables stored in the primary Vertica database.

 **Note** A sync node that uses Vertica Writer must have at least the permission to execute the `INSERT INTO` statement. Whether other permissions are required depends on the SQL statements specified in the `preSql` and `postSql` parameters when you configure the node.

- Vertica Writer does not support the `writeMode` parameter.
- Vertica Writer accesses a Vertica database by using the Vertica database driver. Confirm the compatibility between the driver version and your Vertica database. Vertica Writer uses the following version of the Vertica database driver:

```
<dependency>
  <groupId>com.vertica</groupId>
  <artifactId>vertica-jdbc</artifactId>
  <version>7.1.2</version>
</dependency>
```

Parameters

Parameter	Description	Required	Default value
<code>datasource</code>	The connection name. It must be the same as the name of the added connection. You can add connections in the code editor.	Yes	None
<code>jdbcUrl</code>	<p>The JDBC URL for connecting to the Vertica database. You do not need to set this parameter because the system automatically obtains the value from the connection parameter.</p> <ul style="list-style-type: none"> • You can configure only one JDBC URL for a database. Vertica Writer cannot write data to a database with multiple primary databases. • The format must be in accordance with Vertica official specifications. You can also specify the information of the attachment facility. Example: <code>jdbc:vertica://127.0.0.1:3306/database</code>. 	Yes	None
<code>username</code>	The username that you can use to connect to the database.	Yes	None
<code>password</code>	The password that you can use to connect to the database.	Yes	None

Parameter	Description	Required	Default value
table	The names of the destination tables, which are described in a JSON array. <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> ? Note You do not need to set this parameter because the system automatically obtains the value from the connection parameter. </div>	Yes	None
column	The columns in the destination table to which data is written. Separate the columns with a comma (,), for example, <code>"column": ["id","name","age"]</code> .	Yes	None
preSql	The SQL statement to execute before the sync node is run. Use <code>@table</code> to specify the name of the destination table in the SQL statement. When you execute this SQL statement, DataWorks replaces <code>@table</code> with the name of the destination table.	No	None
postSql	The SQL statement to execute after the sync node is run.	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the Vertica database over the network, and increase the throughput. However, an excessively large value may lead to the OOM error during the data synchronization process.	No	1,024

Configure Vertica Writer by using the codeless UI

The codeless UI is not supported for Vertica Writer.

Configure Vertica Writer by using the code editor

In the following code, a node is configured to write data to a Vertica database.

```
{
  "type": "job",
  "version": "2.0", // The version number.
  "steps": [
    {
      "stepType": "stream",
      "parameter": {},
      "name": "Reader",
      "category": "reader"
    },
    {
      "stepType": "vertica", // The writer type.
      "parameter": {
        "datasource": "The connection name.",
        "username": "",
        "password": "",
        "column": [ // The columns to which data is written.
          "id",
          "name"
        ],
        "connection": [
```

```

    {
      "table": [// The name of the destination table.
        "vertica_table"
      ],
      "jdbcUrl": "jdbc:vertica://ip:port/database"
    }
  ],
  "preSql": [// The SQL statement to execute before the sync node is run.
    "delete from @table where db_id = -1"
  ],
  "postSql": [// The SQL statement to execute after the sync node is run.
    "update @table set db_modify_time = now() where db_id = 1"
  ]
},
"name": "Writer",
"category": "writer"
}
],
"setting": {
  "errorLimit": {
    "record": "0" // The maximum number of dirty data records allowed.
  },
  "speed": {
    "throttle": false, // Specifies whether to enable bandwidth throttling. A value of false indicates that the bandwidth is not throttled. A value of true indicates that the bandwidth is throttled. The maximum transmission rate takes effect only if you set this parameter to true.
    "concurrent": 1 // The maximum number of concurrent threads.
  }
},
"order": {
  "hops": [
    {
      "from": "Reader",
      "to": "Writer"
    }
  ]
}
}
}

```

2.4.5.4.26. Configure Gbase8a Writer

This topic describes the implementation principle and parameter configurations of Gbase8a Writer.

Gbase8a Writer allows you to write data to tables stored in Gbase8a databases. At the underlying implementation level, Gbase8a Writer connects to a remote Gbase8a database through the JDBC Driver and runs the relevant SQL statements to write data to the Gbase8a database.

 **Note** You must configure a connection before configuring Gbase8a Writer.

Parameters

Parameter	Description	Required	Default value
-----------	-------------	----------	---------------

Parameter	Description	Required	Default value
datasource	The connection name. It must be identical to the name of the added connection. You can add connections in the code editor.	Yes	None
table	The name of the table to be synchronized.	Yes	None
writeMode	<p>The write mode. Valid values: <i>insert into</i>, <i>on duplicate key update</i>, and <i>replace into</i>.</p> <ul style="list-style-type: none"> <i>insert into</i>: If a primary key conflict or unique index conflict occurs, data cannot be written to the conflicting rows and is regarded as dirty data. <i>on duplicate key update</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, specified fields in original rows are updated. <i>replace into</i>: If no primary key conflict or unique index conflict occurs, the action is the same as that of insert into. If a conflict occurs, original rows are deleted and new rows are inserted. That is, all fields of original rows are replaced. 	No	<i>insert</i>
column	The columns in the destination table to which data is written. Separate the columns with a comma (.). Example: <code>"column":["id", "name", "age"]</code> . Set the value to an asterisk (*) if data is written to all the columns in the destination table. That is, set the column parameter as follows: <code>"column":["*"]</code> .	Yes	None
preSql	<p>The SQL statement to run before the sync node is run. For example, you can clear outdated data before data synchronization. Currently, you can run only one SQL statement on the codeless user interface (UI), and multiple SQL statements in the code editor.</p> <div style="background-color: #e0f2f1; padding: 5px;"> <p> Note If you specify multiple SQL statements in the code editor, the system does not guarantee that they are run in the same transaction.</p> </div>	No	None
postSql	<p>The SQL statement to run after the sync node is run. For example, you can add a timestamp after data synchronization. Currently, you can run only one SQL statement on the codeless UI, and multiple SQL statements in the code editor.</p> <div style="background-color: #e0f2f1; padding: 5px;"> <p> Note If you specify multiple SQL statements in the code editor, the system does not guarantee that they are run in the same transaction.</p> </div>	No	None
batchSize	The number of data records to write at a time. Setting this parameter can greatly reduce the interactions between Data Integration and the Gbase8a database over the network, and increase the throughput. However, an excessively large value may lead to the out of memory (OOM) error during the data synchronization process.	No	1024

Configure Gbase8a Writer by using the codeless UI

Currently, the codeless UI is not supported for Gbase8a Writer.

Configure Gbase8a Writer by using the code editor

In the following code, a node is configured to write data to the Gbase8a database. For more information about the parameters, see the preceding parameter description.

```
{
  "type":"job",
  "version":"2.0",// The version number.
  "steps":[
    {
      "stepType":"stream",
      "parameter":{},
      "name":"Reader",
      "category":"reader"
    },
    {
      "stepType":"gbase8a",// The writer type.
      "parameter":{
        "postSql":[// The SQL statement to run after the sync node is run.
          "datasource":"","// The connection name.
          "column":[// The columns to be synchronized.
            "id",
            "value"
          ],
          "writeMode":"insert",// The write mode.
          "batchSize":1024,// The number of data records to write at a time.
          "table":"","// The name of the table to be synchronized.
          "preSql":[// The SQL statement to run before the sync node is run.
        ],
        "name":"Writer",
        "category":"writer"
      }
    },
    "setting":{
      "errorLimit":{// The maximum number of dirty data records allowed.
        "record":"0"
      },
      "speed":{
        "throttle":false,// Specifies whether to enable bandwidth throttling.
        "concurrent":1,// The maximum number of concurrent threads.
      }
    },
    "order":{
      "hops":[
        {
          "from":"Reader",
          "to":"Writer"
        }
      ]
    }
  ]
}
```

2.4.5.5. Optimize synchronization performance

This topic describes how to maximize the synchronization speed by adjusting the concurrency configuration, the difference between nodes that are configured with bandwidth throttling and those that are not, and precautions for custom resource groups.

Data Integration is a one-stop platform that supports real-time and offline data synchronization between any connections in any location and in any network environment. You can synchronize 10 TB of data between various types of cloud storage and local storage each day.

DataWorks provides excellent data transmission performance and supports data exchanges between more than 400 pairs of disparate connections. These features allow you to focus on the key issues on constructing big data solutions.

Factors affecting the speed of data synchronization

The factors that affect the speed of data synchronization are listed as follows:

- Source
 - Database performance: the performance of the CPU, memory module, SSD, network, and hard disk.
 - Concurrency: A high concurrency results in a heavy database workload.
 - Network: the bandwidth (throughput) and speed of the network. Generally, a database with better performance can support more concurrent nodes and a larger concurrency value can be set for sync nodes.
- Sync node
 - Synchronization speed: whether an upper limit is set for the synchronization speed.
 - Concurrency: a maximum number of concurrent threads to read data from the source and write data to destination data storage within a single sync node.
 - Nodes that are waiting for resources.
 - Bandwidth throttling: The bandwidth of a single thread is 1,048,576 bit/s. Timeout occurs when the business is sensitive to the network speed. We recommend that you set a smaller value.
 - Whether to create an index for query statements.
- Destination
 - Performance: the performance of the CPU, memory module, SSD, network, and hard disk.
 - Load: Excessive load in the destination database affects the write efficiency within the sync nodes.
 - Network: the bandwidth (throughput) and speed of the network.

You need to monitor and optimize the performance, load, and network of the source and destination databases. The following describes the optimal settings of a sync node.

Concurrency

You can configure the concurrency for a node on the codeless user interface (UI). The following is an example of how to configure the concurrency in the code editor:

```
"setting": {  
  "speed": {  
    "concurrent": 10  
  }  
}
```

Bandwidth throttling

By default, bandwidth throttling is disabled. In a sync node, data is synchronized at the maximum speed given the concurrency configured for the node. Considering that excessively fast synchronization may overstress the database and thus affect the production, Data Integration allows you to limit the synchronization speed and optimize the configuration as required. If bandwidth throttling is enabled, we recommend that you limit the maximum speed to 30 Mbit/s. The following is an example for configuring an upper limit for synchronization speed in the code editor, in which the transmission bandwidth is 1 Mbit/s:

```
"setting": {
  "speed": {
    "throttle": true // The bandwidth throttling is enabled.
    "mbps": 1, // The synchronization speed.
  }
}
```

Note

- When the throttle parameter is set to false, throttling is disabled, and you do not need to configure the mbps parameter.
- The bandwidth value is a Data Integration metric and does not represent the actual network interface card (NIC) traffic. Generally, the NIC traffic is two to three times of the channel traffic, which depends on the serialization of the data storage system.
- A semi-structured file does not have shard keys. If multiple files exist, you can set the maximum job speed to increase the synchronization speed. However, the maximum job speed is limited by the number of files. For example, the maximum job speed limit is set to n Mbit/s for n files. If you set the limit to n+1 Mbit/s, the synchronization speed remains at n Mbit/s. If you set the limit to n-1 Mbit/s, the synchronization is performed at n-1 Mbit/s.
- A table can be partitioned according to the preset maximum job speed only when a maximum job speed and a shard key are configured for a relational database. Relational databases only support numeric shard keys, while Oracle databases support both numeric and string shard keys.

Scenarios of slow data synchronization

- Scenario 1: Resolve the issue that sync nodes to be run on the default resource group remain waiting for resources.
 - Example

When you test a sync node in DataWorks, the node remains waiting for resources and an internal system error occurs.

For example, a sync node is configured to synchronize data from RDS to MaxCompute. The node has waited for about 800 seconds before it is run successfully. However, the log shows that the node runs for only 18 seconds and then stops. The sync node uses the default resource group. When you run other sync nodes, they also remain in the waiting state.

The log is displayed as follows:

```
2017-01-03 07:16:54 : State: 2(WAIT) | Total: 0R 0B | Speed: 0R/s 0B/s | Error: 0R 0B | Stage: 0.0%
```

- Solution

The default resource group is not exclusively used by a single user. It is used by many projects concurrently, not just two or three nodes for a single user. If such resources are insufficient after you start to run a node, the node needs to wait for resources. In this case, the node is completed 800 seconds after you start running the node, but it only takes 10 seconds for the node to be executed.

To improve the synchronization speed and reduce the waiting time, we recommend that you run sync nodes during off-peak hours. Typically, most sync nodes are run between 00:00 and 03:00.

- Scenario 2:

Accelerate nodes that synchronize data from multiple source tables to the same destination table.

- Example

To synchronize data from tables of multiple data stores to a table, you configure multiple sync nodes to run in sequence. However, the synchronization takes a long time.

- Solution

To launch multiple concurrent nodes that write data to the same destination database, pay attention to the following points:

- Ensure that the destination database can support the execution of all the concurrent nodes.
- You can configure a sync node that synchronizes multiple source tables to the same destination table. Alternatively, you can configure multiple nodes to run concurrently in the same workflow.
- If resources are insufficient, you can configure sync nodes to run during off-peak hours.

- Scenario 3:

If no index is added when the WHERE clause is used, a full table scan slows down the data synchronization.

- Example

SQL statement:

```
select bid,inviter,uid,createTime from `relatives` where createTime>='2016-10-23 00:00:00'and reateTime<'2016-10-24 00:00:00';
```

Assume that the sync node started to run the preceding statement at 2016-10-25 11:01:24 and started to return results from 2016-10-25 11:11:05. It took a long time to finish the sync node.

- Cause

When the WHERE clause is used for a query, the createTime column is not indexed, resulting in a full table scan.

- Solution

We recommend that you use an indexed column or add an index to the column that you want to scan if you use the WHERE clause.

2.4.6. Real-time synchronization

2.4.6.1. Overview

This topic describes the definition, highlights, supported data sources, and architecture of the real-time synchronization feature. This topic also describes how to use this feature.

Definition

The real-time synchronization feature allows you to synchronize the data changes in a source database to a destination database in real time. For example, after you add, modify, or delete the data in a source database, the real-time synchronization feature synchronizes these changes to a destination database in real time. If a real-time synchronization is performed during a batch synchronization for full historical data, all the data in your source database is synchronized to your destination database. The real-time synchronization feature helps ensure that the data in your destination database is updated in real time and consistent with that in your source database.

Highlights

- Diverse data sources
 - Star-shaped combination is supported. Each supported source can be combined with a supported destination to synchronize data.
 - The feature will soon allow you to synchronize data from a single source to multiple destinations at the same time.
- Comprehensive synchronization solutions
 - You can configure a synchronization solution to synchronize the full data or incremental data from a database to MaxCompute or Hologres.
 - The feature synchronizes full data first and then continuously synchronizes incremental data to the destination database based on the synchronization solution that you configure.
- Real-time data extraction

The feature can extract data from the table shards, a single table, and multiple tables in a database, and extract the data definition language (DDL) statements of tables.
- Data processing

The feature can filter data and replace strings during synchronization.
- Monitoring and alerting
 - The feature monitors the service latency, failovers, dirty data, heartbeats, and failures during synchronization.
 - Alert notifications can be sent by email, phone call, and DingTalk message.
- Small impact on the source

The feature is optimized to have a small impact on the source.
- Graphical development
 - You can develop real-time synchronization nodes in a graphical user interface (GUI) without the need to write code.
 - The feature is easy to use even for beginners.

Supported data sources

- Source: MySQL Binlog, Oracle CDC, Kafka, DataHub, LogHub, and PolarDB.
- Destination: MaxCompute, Hologres, Kafka, and DataHub.
- The feature can filter data and replace strings during synchronization.

How to use the feature

You can use the real-time synchronization feature to create real-time synchronization solutions or synchronization nodes.

- For more information about how to create a real-time synchronization node, see [Create a real-time synchronization node](#).

- For more information about how to create a synchronization solution, see [Go to the Sync Solutions page](#).

2.4.6.2. Data sources supported by real-time synchronization

Real-time synchronization supports reader, writer, and conversion plug-ins. The following table lists the plug-ins.

Category	Plug-in	References
Reader	MySQL Binlog Reader	MySQL binlogs
	Oracle Change Data Capture (CDC) Reader	Oracle CDC
	DataHub Reader	DataHub
	LogHub Reader	LogHub
	Kafka Reader	Kafka
Writer	Hologres Writer	Hologres
	DataHub Writer	DataHub
	Kafka Writer	Kafka
	MaxCompute Writer	Configure MaxCompute Writer
Conversion	Data filtering	Data filter
	Groovy	Groovy
	String replacement	String replacement

 **Note** A real-time synchronization node cannot be directly run on the node configuration tab. Instead, you must save and commit the node before the real-time synchronization node is run in the production environment.

Basic configuration

After you configure the reader, writer, and conversion plug-ins, you can click the **Basic configuration** panel in the right-side navigation pane to configure the synchronization node.

Parameter	Description
Description	The description of the real-time synchronization node.
JVM parameters	The Java virtual machine (JVM) memory allocated for the real-time synchronization node. If this parameter is not specified, Data Integration automatically allocates JVM memory based on your node configurations.

2.4.6.3. Create, configure, commit, and manage real-time synchronization nodes

DataWorks allows you to synchronize data in real time. This topic describes how to create, configure, commit, and manage real-time synchronization nodes.

Create a real-time synchronization node

1. [Log on to the DataWorks console.](#)
2. On the DataStudio page, move the pointer over the  icon and choose **Data Integration > Real-time synchronization**. Alternatively, you can find the required workflow, right-click the workflow name, and then choose **Create > Data Integration > Real-time synchronization**.
3. In the **Create Node** dialog box, configure the parameters.

Parameter	Description
Node Type	The type of the node. Default value: Real-time synchronization .
Sync Method	The method used to synchronize data. Valid values: <ul style="list-style-type: none"> ◦ End-to-end ETL: synchronizes data in one table to one or more tables. Data type conversion is supported during the synchronization. ◦ Migration to Hologres: synchronizes all or some tables in a database to Hologres. Destination tables can be automatically created in Hologres. ◦ Migration to MaxCompute: synchronizes all or some tables in a database to MaxCompute. ◦ Migration to DataHub: synchronizes all or some tables in a database to DataHub.
Node Name	The name of the node. The name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).
Location	The directory in which the real-time synchronization node is stored.

4. Click **Commit**.

Configure the real-time synchronization node

The operations that you can perform on the configuration tab of the real-time synchronization node vary based on the synchronization method that you selected.

- To configure the real-time synchronization node for which **Sync Method** is set to **End-to-end ETL**, perform the following steps:
 - i. Click **Basic configuration** in the right-side navigation pane. In the Basic configuration panel, select the required resource group from the **Resource Group** drop-down list.

No.	Description
1	The left-side navigation tree, which consists of the Input , Output , and Conversion sections.
2	The configuration canvas of the real-time synchronization node. You can drag a component from the navigation tree to the canvas and configure the component.

No.	Description
3	The property configuration panel of the real-time synchronization node. This panel appears after you click Basic configuration in the right-side navigation pane.

- ii. Drag components from the navigation tree to the canvas, and draw lines to connect the components. This way, the components synchronize data based on the connections.
 - iii. Click each component on the canvas. In the panel that appears, configure the parameters.
 - iv. Click the  icon in the top toolbar.
- To configure the real-time synchronization node for which **Sync Method** is set to **Migration to Hologres**, perform the following steps:
 - i. Click **Basic configuration** in the right-side navigation pane. In the Basic configuration panel, select the required resource group from the **Resource Group** drop-down list.

 **Notice** You must select a resource group before you commit the node. Otherwise, the system returns an error when you commit the node.

- ii. In the **Data source** section, specify **Type** and **Data source**.
- iii. In the **Select the source table for synchronization** section, select the tables that you want to synchronize in the **SOURCE Table** list and click the  icon to move the tables to the **Selected Source table** list.

The **SOURCE Table** list displays all the tables in the source. You can select all or some tables to synchronize them at a time.

 **Notice** If a selected table does not have a primary key, the table cannot be synchronized in real time.

- iv. (Optional) In the **Set synchronization rules** section, click **Add rule** and select an option to configure naming rules for destination tables.

Supported options include **Table name conversion rules** and **Target table name rule**.

- **Table name conversion rules:** the rule for converting the names of source tables to those of destination tables.
- **Target table name rule:** the rule for adding a prefix and suffix to the converted names of destination tables.

- v. Click **Next Step**.
- vi. In the **Set target table** step, specify **Target Hologres data source** and **Schema**.
- vii. Click **Reload source table and Hologres Table mapping** to configure the mappings between the source tables and destination Hologres tables.
- viii. View the mapping progress, source tables, and mapped destination tables, and click **Next Step**.

You can view the mapping progress between the source tables and destination tables. The mapping may take a long period of time if you want to synchronize a large number of tables.

An error message appears if the selected source table does not have a primary key. The synchronization can be performed if one of the selected source tables has a primary key. Source tables without primary keys are ignored during the synchronization.

You can set Table creation method to **Create tables automatically** or **Use existing Table**. The name of the destination table that appears in the **Hologres Table name** column varies based on the setting of Table creation method.

- If you set **Table creation method** to **Create tables automatically**, the name of the destination table that is automatically created appears. You can click the table name to view and modify the table creation statements.
 - If you set **Table creation method** to **Use existing Table**, you must select a table from the drop-down list in the **Hologres Table name** column.
- ix. In the **Run resource settings** step, specify **Maximum number of connections supported by source read** and **Number of concurrent writes on the target side**. Then, click the  icon in the top toolbar.
- To configure the real-time synchronization node for which **Sync Method** is set to **Migration to MaxCompute**, perform the following steps:
 - i. Click **Basic configuration** in the right-side navigation pane. In the Basic configuration panel, select the required resource group from the **Resource Group** drop-down list.
 - ii. In the **Data source** section, specify **Type** and **Data source**.
 - iii. In the **Select the source table for synchronization** section, select the tables that you want to synchronize in the **SOURCE Table** list and click the  icon to move the tables to the **Selected Source table** list.

The **SOURCE Table** list displays all the tables in the source. You can select all or some tables to synchronize them at a time.

 **Notice** If a selected table does not have a primary key, the table cannot be synchronized in real time.

- iv. In the **Set synchronization rules** section, click **Add rule** and select an option to configure naming rules for destination tables.
- Supported options include **Table name conversion rules** and **Target table name rule**.
- **Table name conversion rules**: the rule for converting the names of source tables to those of destination tables.
 - **Target table name rule**: the rule for adding a prefix and suffix to the converted names of destination tables.
- v. Click **Next Step**.
- vi. In the **Set target table** step, select a data source from the **Target MaxCompute data source** drop-down list and click the  icon next to **MaxCompute time automatic partition settings**. In the **Edit** dialog box, set the partition interval of tables in MaxCompute to day or hour.
- vii. Click **Reload source table and MaxCompute Table mapping** to configure the mappings between the source tables and destination MaxCompute tables.
- viii. View the mapping progress, source tables, and destination tables. Then, click **Next Step**.

You can view the mapping progress between the source tables and destination tables. The mapping may take a long period of time if you want to synchronize a large number of tables.

An error message appears if the selected source table does not have a primary key. The synchronization can be performed if one of the selected source tables has a primary key. Source tables without primary keys are ignored during the synchronization.

You can set Table creation method to **Create tables automatically** or **Use existing Table**. The name of the destination table that appears in the **MaxCompute Table name** column varies based on the setting of Table creation method.

- If you set **Table creation method** to **Create tables automatically**, the name of the destination table that is automatically created appears. You can click the table name to view and modify the table creation statements.
 - If you set **Table creation method** to **Use existing Table**, you must select a table name from the drop-down list in the **MaxCompute Table name** column.
- ix. In the **Run resource settings** step, specify **Maximum number of connections supported by source read** and **Number of concurrent writes on the target side**. Then, click the  icon in the top toolbar.
- To configure the real-time synchronization node for which **Sync Method** is set to **Migration to DataHub**, perform the following steps:
 - i. On the node configuration tab that appears, click **Basic configuration** in the right-side navigation pane. In the Basic configuration panel, select the required resource group from the **Resource Group** drop-down list.
 - ii. In the **Data source** section, specify **Type** and **Data source**.
 - iii. In the **Select the source table for synchronization** section, select the tables that you want to synchronize in the **SOURCE Table** list and click the  icon to move the tables to the **Selected Source table** list.

The **SOURCE Table** list displays all the tables in the source. You can select all or some tables to synchronize them at a time.

 **Notice** If a selected table does not have a primary key, the table cannot be synchronized in real time.

- iv. In the **Set synchronization rules** section, click **Add rule** and then select an option to configure naming rules for destination DataHub topics.
- Supported options include **SOURCE table name and Topic conversion rules** and **Target Topic rules**.
- v. Click **Next Step**.
- vi. In the **Set target table** step, select a data source from the **Target DataHub data source** drop-down list and then click **Reload source table and DataHub Topic mapping** to configure the mappings between the source tables and destination DataHub topics.
- vii. View the mapping progress, source tables, and destination topics. Then, click **Next Step**.

You can view the mapping progress between the source tables and destination topics. The mapping may take a long period of time if you want to synchronize a large number of tables.

You can set Topic creation method to **Create tables automatically** or **Use existing Topic**. The message that appears in the **DataHub Topic** column varies based on the setting of Topic creation method.

- If you set Topic creation method to **Create tables automatically**, the **Create tables automatically** dialog box appears after you click **Next Step**. Click **Start table building** in the dialog box, and then click **Close** after the topic is created.
 - If you set Topic creation method to **Use existing Topic**, you must select a topic from the drop-down list in the **DataHub Topic** column.
- viii. In the **Run resource settings** step, specify **Maximum number of connections supported by**

source read and **Number of concurrent writes on the target side**. Then, click the  icon in the top toolbar.

Commit the real-time synchronization node

1. On the configuration tab of the real-time synchronization node, click the  icon in the top toolbar.
2. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
3. Click **OK**. If the workspace that you use is in standard mode, you must click **Deploy** in the upper-right corner after you commit the real-time synchronization node.

Manage the real-time synchronization node

1. After you commit or deploy the real-time synchronization node, click **Operation Center** in the upper-right corner of the DataStudio page to manage the node on the **Real Time DI** page.
2. On the **Real Time DI** page, find the real-time synchronization node, click the node name, and then view the O&M details about the node. On the Real Time DI page, you can start, stop, undeploy, or configure alert settings for the real-time synchronization node.
 - o To start a node that is not running, perform the following steps:
 - a. Find the node and click **Start** in the Operation column.
 - b. In the **Start** dialog box, configure the parameters.

Parameter	Description
Whether to reset the site	Specifies whether to set the time point for the next startup. If you select Reset site , the Start time point and Time zone parameters are required.
Start time point	The date and time for starting the real-time synchronization node.
Time zone	The time zone where the source resides. Select a time zone from the Time zone drop-down list.
Failover	<ul style="list-style-type: none"> ■ The condition for automatically terminating the real-time synchronization node. You can specify the maximum number of dirty data records allowed. If you set this parameter to 0, no dirty data records are allowed. If this parameter is not specified, the node continues to run no matter whether dirty data records exist. ■ You can also specify the maximum number of failover times. If you do not specify the times, the node is automatically terminated if the node fails 100 times within 5 minutes. This prevents resource occupation caused by frequent startups.

- c. Click **OK**.
- o To stop a running node, perform the following steps:
 - a. Find the node and click **Stop** in the Operation column.
 - b. In the message that appears, click **Stop**.
- o To undeploy a node that is not running, perform the following steps:
 - a. Find the node and click **Offline** in the Operation column.
 - b. In the message that appears, click **Offline**.

- Find the node and click **Alarm settings** in the Operation column. Then, you can view alert event information and alert rules on the **Alert event** and **Alarm rules** tabs.
- To configure alert settings for a node, perform the following steps:
 - a. Select the node and click **New Alarm** in the lower part of the page.
 - b. In the **New rule** dialog box, configure the parameters.

Parameter	Description
Name	Required. The name of the rule that you want to create.
Description	The description of the rule.
Indicators	The metrics in the rule that you want to create. Valid values: Task Status , Business latency , Failover , Dirty Data , and DDL error .
Threshold	The threshold for reporting an alert. The default value is 5 minutes for both WARNING and CRITICAL alerts.
Alarm interval	The interval at which an alert is reported. The default value is 5 minutes.
WARNING	The method used to send alert notifications. The value of this parameter can be only Mail .
CRITICAL	
Recipient	The alert recipient. Select a recipient from the Receiver drop-down list.

- c. Click **OK**.
- To modify alert settings for a node, perform the following steps:
 - a. Select the node whose alarm settings you want to modify and click **Operation alarm** in the lower part of the page.
 - b. In the **Operation alarm** dialog box, specify **Operation type** and **Alarm indicators**.
DataWorks automatically modifies all the rules for the selected alert types at a time.
 - c. Click **OK**.

2.4.6.4. Reader

2.4.6.4.1. MySQL binlogs

The MySQL binlog reader reads data from the tables of MySQL databases in real time.

The MySQL binlog reader reads data from the MySQL database in real time by using Canal, which parses incremental MySQL binlogs and subscribes to data changes.

Parameters

Parameter	Description	Required	Default value
-----------	-------------	----------	---------------

Parameter	Description	Required	Default value
dbHost	The endpoint of the database. You can set this parameter to the domain name or the IP address.	Yes	None
dbName	The name of the database.	Yes	None
port	The port number of the database.	Yes	None
table	The name of the table to be synchronized.	Yes	None
username	The username used to access the database.	Yes	None
password	The password used to access the database.	Yes	None
startTimestampMills	<p>The start time of data synchronization. The value of this parameter is a 13-bit timestamp.</p> <div style="background-color: #e6f2ff; padding: 5px;"> <p>Note If the start time you specify is later than the last data change time of MySQL binlogs, data synchronization starts from the last data change time of MySQL binlogs.</p> <p>If the start time you specify is earlier than the earliest data change time of MySQL binlogs, no data is synchronized.</p> </div>	No	None

Example

In the following code, a node is configured to read data from a MySQL database and write data to DataHub in real time:

```
{
  "order": {
    "hops": [
      {
        "from": "mysqlbinlog_01",
        "to": "datahub_01"
      }
    ]
  }
}
```

```
}
]
},
"setting": {
  "errorLimit": {
    "record": 0
  },
},
"steps": [
  {
    "category": "reader",
    "name": "mysqlbinlog_01",
    "parameter": {
      "password": "xxx",
      "column": [
        "_log_file_name_offset_",
        "_operation_type_",
        "_execute_time_",
        "_before_image_",
        "_after_image_",
        "id",
        "pipeline_name",
        "execute_name",
        "context"
      ],
      "dbHost": "127.0.0.1",
      "dbName": "xxx",
      "port": 3306,
      "startTimestampMills": 1555689600000,
      "table": "xxx",
      "username": "xxx"
    },
    "stepType": "mysqlbinlog"
  },
  {
    "category": "writer",
    "name": "datahub_01",
    "parameter": {
      "accessKey": "xxx",
      "accessId": "xxx",
      "batchSize": 1000,
      "column": [
        "_log_file_name_offset_",
        "_execute_time_",
        "_before_image_",
        "_after_image_",
        "id",
        "pipeline_name",
        "execute_name",
        "context"
      ],
      "columnMapping": [
        {
          "dstColName": "_log_file_name_offset_",
          "sourceColName": "_log_file_name_offset_"
        },
        {
          "dstColName": "_execute_time_"
        }
      ]
    }
  }
]
```

```

      "dstColName": "_execute_time_",
      "sourceColName": "_execute_time_"
    },
    {
      "dstColName": "_before_image_",
      "sourceColName": "_before_image_"
    },
    {
      "dstColName": "_after_image_",
      "sourceColName": "_after_image_"
    },
    {
      "dstColName": "id",
      "sourceColName": "id"
    },
    {
      "dstColName": "pipeline_name",
      "sourceColName": "pipeline_name"
    },
    {
      "dstColName": "execute_name",
      "sourceColName": "execute_name"
    },
    {
      "dstColName": "context",
      "sourceColName": "context"
    }
  ],
  "endpoint": "xxx",
  "project": "xxx",
  "topic": "xxx"
},
"stepType": "datahub"
}
]
}

```

2.4.6.4.2. Oracle CDC

StreamX uses Change Data Capture (CDC) to synchronize data of Oracle databases in real time. This synchronization mode uses triggers on the source database to capture change data. When the CDC synchronization mode is enabled, the performance of the database may be deteriorated.

Before synchronizing data from an Oracle database, you need to perform a series of operations in the database. Take data synchronization between two Oracle tables as an example.

Synchronize data from the source table named `synctest.trade` to the destination table named `cdcuser.trade_target`.

The source and destination tables are created by using the `CREATE TABLE` statement. StreamX obtains changes of the incremental data and synchronizes the changed data to the destination table. The changed incremental data that StreamX reads includes the specific information about the action types. The `UPDATE` statements are divided into the before image and after image. The before image contains the data before the update. The after image contains the updated data.

Procedure

1. Create a table. You can also select an existing table.

```
create tablespace ts_cdcpub  
logging datafile '/u01/app/oracle/oradata/mydb/cdcpub01.dbf'  
size 5000M autoextend off;
```

2. Create a user named cdcuser to store incremental data and grant permissions to the user.

```
CREATE USER cdcuser IDENTIFIED BY cdcuser DEFAULT TABLESPACE ts_cdcpub QUOTA UNLIMITED ON ts_cdcpub;  
GRANT CREATE SESSION TO cdcuser;  
GRANT CREATE TABLE TO cdcuser;  
GRANT SELECT_CATALOG_ROLE TO cdcuser;  
GRANT EXECUTE_CATALOG_ROLE TO cdcuser;  
GRANT CONNECT, RESOURCE TO cdcuser;
```

3. Grant the SELECT permission on the table to be synchronized to the cdcuser user.

```
grant select on syntest.trade to cdcuser;
```

4. Create a change set and a change table.

- o Create a change set

```
DBMS_CDC_PUBLISH.CREATE_CHANGE_SET(  
change_set_name => 'tradetest',  
description => 'Change set for syntest.trade info',  
change_source_name => 'SYNC_SOURCE');
```

- o Create a change table

```
DBMS_CDC_PUBLISH.CREATE_CHANGE_TABLE(  
owner => 'cdcuser',  
change_table_name => 'trade_ct',  
change_set_name => 'tradetest',  
source_schema => 'syntest',  
source_table => 'trade',  
column_type_list => 'id number,money number,op_user varchar(100)',  
capture_values => 'both',  
rs_id => 'y',  
row_id => 'n',  
user_id => 'n',  
timestamp => 'n',  
object_id => 'n',  
source_colmap => 'y',  
target_colmap => 'y',  
DDL_MARKERS => 'N',  
options_string => 'TABLESPACE ts_cdcpub');
```

5. After the change set and change table are created, add an Oracle CDC connection and an Oracle connection.
6. Go to the **Data Integration** page. In the left-side navigation pane, choose **Nodes > Real-Time Sync**.
 - o Configure the reader. The reader currently supports four connection types including MySQL Binlog, Oracle CDC, DataHub, and LogHub. Configure the corresponding connection before configuring the reader.

Parameter	Description
Connection	The name of the connection.

Parameter	Description
Table	By default, four data records in the selected table are available for preview.
Start Offset	The time when the data synchronization node runs.
Time Zone	The time zone of the data synchronization node. The time zone can be determined based on the time zone of the connection.

- Configure the writer. The writer currently supports three connection types including MySQL, Oracle, and DataHub. Configure the corresponding connection before configuring the writer.

Specify and map the fields in the source and destination.

2.4.6.4.3. DataHub

The DataHub stream reader reads data from DataHub in real time by using the DataHub SDK.

The reader keeps running after it is started and reads data from DataHub when DataHub stores new data. The DataHub stream reader has the following two features:

- Reads data in real time.
- Reads data concurrently based on the number of shards in DataHub.

Parameters

Parameter	Description	Required
endpoint	The endpoint used to access DataHub.	Yes
accessId	The AccessKey ID used to access DataHub.	Yes
project	The destination project of DataHub. A project is the resource management unit in DataHub for resource isolation and control.	Yes
topic	The destination topic of DataHub.	Yes
batchSize	The size of data read at a time.	No. Default value: 1024.
startTimestampMills	The start time of data consumption. The time is an integer accurate to milliseconds. Example: 1554739200000.	No. You must specify either the startTimestampMills parameter or the position parameter.
position	The start time of data consumption for each shard. You can specify this parameter for each shard. Example: <code>{"allShardTimestampMills": {"0":1549098048000,"1":1549098048000,"2":1549098048000,"3":1549098048000,"4":1549098048000}}</code> .	No. You must specify either the startTimestampMills parameter or the position parameter.

Example

In the following code, a node is configured to read data from DataHub and write data to a MySQL database in real time:

```
{
  "connections": [],
  "name": "yj_datahub2datahub4",
  "order": {
    "hops": [
      {
        "from": "datahubstreamreader",
        "to": "mysqloutput001"
      }
    ]
  },
  "params": [],
  "setting": {
    "performance": {}
  },
  "steps": [
    {
      "name": "datahubstreamreader",
      "category": "reader",
      "parameter": {
        "endpoint": "http://dh-cn-hangzhou.aliyuncs.com",
        "accessId": "*****",
        "accessKey": "*****",
        "project": "streamx_datahub",
        "topic": "mysqlbinlog_to_datahub_topic2",
        "batchSize": 512,
        "startTimestampMills": 1554739200000,
        "position": "{\"allShardTimestampMills\": {\"0\":1549098048000,\"1\":1549098048000,\"2\":1549098048000,\"3\":1549098048000,\"4\":1549098048000}}",
        "column": [
          "cdc_record_id",
          "cdc_operation",
          "cdc_timestamp",
          "cdc_before_image",
          "cdc_after_image",
          "id",
          "name",
          "age"
        ]
      },
      "stepType": "datahubstream"
    },
    {
      "name": "mysqloutput001",
      "category": "writer",
      "parameter": {
        "batchSize": 100,
        "writeMode": "insert",
        "username": "*****",
        "password": "*****",
        "column": [
          "cdc_record_id",
          "cdc_operation"
        ]
      }
    }
  ]
}
```

```

cdc_operation",
"cdc_timestamp",
"cdc_before_image",
"cdc_after_image",
"id",
"name",
"age"
],
"connection": [
{
"jdbcUrl": "jdbc:mysql://10.101.83.3:3306/test",
"table": [
"streamx_test_real_time_inc_datahub"
]
}
]
},
"stepType": "mysql"
}
],
"type": "job",
"version": "2.0"
}

```

2.4.6.4.4. LogHub

The LogHub stream reader reads data from LogHub in real time by using the LogHub SDK.

The reader reads data from LogHub topics you specified in real time and supports shard merge and split.

 **Note** When shard merge or split is enabled, duplicate data records may exist. However, data missing is avoided.

Parameters

Parameter	Description	Required	Default value
endpoint	The endpoint used to access LogHub.	Yes	None
project	The name of the LogHub project.	Yes	None
logstore	The Logstore of LogHub.	Yes	None
table	The name of the table to be synchronized.	Yes	None
accessId	The AccessKey ID used to access LogHub.	Yes	None
accessKey	The AccessKey secret used to access LogHub.	Yes	None

Parameter	Description	Required	Default value
startTimestampMills	<p>The start time of data synchronization. The value of this parameter is a 13-bit timestamp.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note If the start time you specify is later than the last data change time of MySQL binlogs, data synchronization starts from the last data change time of MySQL binlogs. If the start time you specify is earlier than the earliest data change time of MySQL binlogs, no data is synchronized.</p> </div>	No	None

Example

In the following code, a node is configured to read data from LogHub and write data to DataHub in real time:

```
{
  "order": {
    "hops": [
      {
        "from": "loghubstream_01",
        "to": "datahub_01"
      }
    ]
  },
  "steps": [
    {
      "category": "reader",
      "name": "loghubstream_01",
      "parameter": {
        "accessKey": "<yourAccessKey>",
        "accessId": "<yourAccessId>",
        "column": [
          "col1",
          "col2",
          "col3"
        ],
        "endpoint": "<yourEndpoint>",
        "logstore": "<yourLogstore>",
        "project": "<yourProject>",
        "startTimestampMills": 1555050358000,
      },
      "stepType": "loghubstream"
    }
  ],
}
```

```

{
  "category": "writer",
  "name": "datahub_01",
  "parameter": {
    "accessKey": "<yourAccessKey>",
    "batchSize": 1000,
    "column": [
      "col1",
      "col2",
      "col3"
    ],
    "columnMapping": [
      {
        "dstColName": "col1",
        "sourceColName": "col1"
      },
      {
        "dstColName": "col2",
        "sourceColName": "col2"
      },
      {
        "dstColName": "col3",
        "sourceColName": "col3"
      }
    ],
    "endpoint": "<yourEndpoint>",
    "project": "<yourProject>",
    "topic": "<yourTopic>"
  },
  "stepType": "datahub"
}
]
}

```

2.4.6.4.5. Kafka

Create a real-time sync node

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page.
3. In the left-side navigation pane, choose **Nodes > Real-Time Sync**. Click **Create Node** in the upper-right corner.
4. In the **Create Node** dialog box that appears, set **Node Name** and **Description**.
5. Click **OK**.

Configure an input node of the Kafka type

Drag **Kafka** under **Reader** to the canvas. Click the node and then configure the node in the **Node Settings** dialog box that appears.

Parameter	Description
server	The broker server address of Kafka in the format of <code>ip:port</code> .

Parameter	Description
topic	The name of the Kafka topic.
keyType	The type of the Kafka key.
valueType	The type of the Kafka value.
Consumer Offset	The consumer offset from which Kafka starts to consume data.
Parameters	The JSON-formatted parameters for specifying additional Kafka configurations. Example: group_id.
Start Offset	The date and time of the start offset.
Time Zone	The time zone.
Output Fields	The output fields of the node. You can customize the output fields.

Take the following Kafka configuration script as an example.

```
"parameter": {
  "server": "100.81.127.26:9092",
  "topic": "xin_001",
  "keyType": "long",
  "valueType": "string",
  "startupMode": "earliestOffset", -- Consume data from the earliest consumer offset.
  "discoveryIntervalMillis": 10000, -- The interval between partitions. The value is automatically discovered by the
system.
  "kafkaConfig": {
    "group.id": "xin_group_001"
  }
}
```

2.4.6.4.6. Configure PolarDB Reader

PolarDB Reader can read data only from PolarDB for MySQL databases. It cannot read data from PolarDB for PostgreSQL databases.

Procedure

1. Log on to the DataWorks console.
2. Move the pointer over the  icon and choose **Data Integration > Real-time synchronization**. Alternatively, you can click the required workflow, right-click **Data Integration**, and then choose **Create > Real-time synchronization**.
3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

 **Notice** The node name must be 1 to 128 characters in length. It can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the configuration tab of the real-time synchronization node, drag **PolarDB** in the **Input** section to the canvas on the right.

- Click the **PolarDB** node. In the panel that appears, configure the parameters.

Parameter	Description
Data source	The PolarDB for MySQL data source that you have added. You can select only a PolarDB for MySQL data source. If no data source is available, click New data source to add one on the Data Source page.
Table	The name of the table from which you want to read data. You can click Data preview on the right to preview the selected table.
Output field	The fields from which you want to read data.

- Click the  icon in the toolbar.

2.4.6.5. Writer

2.4.6.5.1. Configure MaxCompute Writer

MaxCompute offers a comprehensive data import solution to support fast computing for large amounts of data.

Prerequisites

A reader or conversion node is configured.

Procedure

- Log on to the DataWorks console.
- Move the pointer over the  icon and choose **Data Integration > Real-time synchronization**. Alternatively, you can click the required workflow, right-click **Data Integration**, and then choose **Create > Real-time synchronization**.
- In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

 **Notice** The node name must be 1 to 128 characters in length. It can contain letters, digits, underscores (_), and periods (.).

- Click **Commit**.
- On the configuration tab of the real-time synchronization node, drag **MaxCompute** in the **Output** section to the canvas on the right. Connect the MaxCompute node to the configured reader or conversion node.
- Click the **MaxCompute** node. In the panel that appears, configure the parameters.

Parameter	Description
Data source	The MaxCompute data source that you have configured. You can select only a MaxCompute data source. If no data source is available, click New data source on the right to add one on the Data Source page.

Parameter	Description
Table	<p>The name of the MaxCompute table to which you want to write data.</p> <p>You can click One-Click table creation on the right to create a table, or click Data preview to preview the selected table.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Notice Before you create a table, connect the MaxCompute node to a reader node, and make sure that the output field parameters are specified for the reader node.</p> </div>
Partition message	The information about the partitioned MaxCompute table.
Field Mapping	The field mappings between the source and destination. Click Field Mapping and configure field mappings. The synchronization node synchronizes data based on the field mappings.

If you want to create a table, click **One-Click table creation** next to Table. In the **New data table** dialog box, configure the parameters.

Parameter	Description
Table name	The name of the MaxCompute table.
Life cycle	The lifecycle of the MaxCompute table.
Data field structure	The field structure of the MaxCompute table. To add a field, click Add .
Partition settings	<p>The partitions of the MaxCompute table.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p> Notice You must configure at least two levels of partitions, which are yearly and monthly partitions. You can configure a maximum of five levels of partitions, which are yearly, monthly, daily, hourly, and minutely partitions.</p> </div>

7. Click the  icon in the toolbar.

2.4.6.5.2. Configure Hologres Writer

You can build a real-time data warehouse by using the real-time write capability of Hologres.

Prerequisites

A reader or conversion node is configured.

Procedure

1. Log on to the DataWorks console.
2. Move the pointer over the  icon and choose **Data Integration > Real-time synchronization**. Alternatively, you can click the required workflow, right-click **Data Integration**, and then choose **Create > Real-time synchronization**.
3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

 **Notice** The node name must be 1 to 128 characters in length. It can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the configuration tab of the real-time synchronization node, drag **Hologres** in the **Output** section to the canvas on the right. Then, draw a line to connect it to the configured reader or conversion node.
6. Click the **Hologres** node. In the panel that appears, configure the parameters.

Parameter	Description
Data source	<p>The Hologres data source that you configured. You can select only a Hologres data source.</p> <p>If no data source is available, click New data source to add one on the Data Source page.</p>
Table	<p>The name of the Hologres table to which you want to write data.</p> <p>You can click One-Click table creation on the right to create a table, or click Data preview to preview the selected table.</p>
Dynamic Time Partition	<p>If the Hologres table is a partitioned table, you must specify a dynamic time-based partition.</p> <p>The dynamic time-based partition parses the value of a source field in the <code>yyyymmddhhmmss</code> format. After the value is parsed, you can use the dynamic partition whose name is a string of variables in the destination table. The destination partition varies based on the value of the source field.</p> <p>For example, the value of the source field is <code>20200816</code>, and the name of the destination partition is in the <code>{yyyy}-{mm}-{dd}</code> format. In this case, the value is written to the <code>2020-08-16</code> partition.</p>
Job type	<p>The type of the data write operation. Valid values: Replay (replay operation log to restore data) and Insert (direct archive save).</p> <ul style="list-style-type: none"> ◦ Replay (replay operation log to restore data): indicates that Hologres Writer performs the same operation on the Hologres destination as that performed on the source. For example, if the <code>INSERT</code> statement is executed to add a record to the source, Hologres Writer executes the <code>INSERT</code> statement to add the same record to the Hologres destination. If the <code>UPDATE</code> or <code>DELETE</code> statement is executed in the source, Hologres Writer executes the <code>UPDATE</code> or <code>DELETE</code> statement in the Hologres destination. ◦ Insert (direct archive save): indicates that Hologres Writer uses the Hologres destination as streaming data storage. Data is synchronized from the source to the Hologres destination by using the <code>INSERT</code> statement.

Parameter	Description
Writer conflict policy	The solution to data write conflicts. Valid values: <ul style="list-style-type: none">◦ Cover (Overwrite): indicates that Hologres Writer uses the new data synchronized from the source to overwrite the existing data in the Hologres destination.◦ Ignore (Ignore): indicates that Hologres Writer ignores the new data synchronized from the source and retains the existing data in the Hologres destination.
Field Mapping	The field mappings between the source and destination. Click Field Mapping and configure field mappings between the source and destination. The synchronization node synchronizes data based on the field mappings.

7. Click the  icon in the toolbar.

2.4.6.5.3. Configure DataHub Writer

DataHub is a platform that is designed to process streaming data. You can publish and subscribe to streaming data in DataHub and distribute the data to other platforms. DataHub allows you to analyze streaming data and build applications based on the streaming data.

Prerequisites

A reader or conversion node is configured.

Context

DataHub Writer writes data to DataHub by using DataHub SDK for Java. The following code provides an example of DataHub SDK for Java:

```
<dependency>
  <groupId>com.aliyun.datahub</groupId>
  <artifactId>aliyun-sdk-datahub</artifactId>
  <version>2.5.1</version>
</dependency>
```

Procedure

1. Log on to the DataWorks console.
2. Move the pointer over the  icon and choose **Data Integration > Real-time synchronization**. Alternatively, you can click the required workflow, right-click **Data Integration**, and then choose **Create > Real-time synchronization**.
3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

 **Notice** The node name must be 1 to 128 characters in length. It can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the configuration tab of the real-time synchronization node, drag **DataHub** in the **Output** section to the canvas on the right. Connect the new node to the configured reader or conversion node.
6. Click the new **DataHub** node. In the configuration panel that appears, set the parameters.

Parameter	Description
Data source	The configured DataHub data source. In this example, you can select only a DataHub data source. If no data source is available, click New data source on the right to add one on the Data Source page.
Topic	The name of the DataHub topic to which you want to write data. You can click Data preview on the right to preview the selected topic.
Batch number	The number of records to write at a time.
Field Mapping	The mappings between fields in the source and destination. DataWorks synchronizes data based on the field mappings.

7. Click the  icon in the toolbar.

2.4.6.5.4. Kafka

Create a real-time sync node

1. Log on to the DataWorks console.
2. Move the pointer over the DataWorks icon in the upper-left corner and select **Data Integration**.
3. In the left-side navigation pane, choose **Nodes > Real-Time Sync**. Click **Create Task** in the upper-right corner.
4. In the **Create Node** dialog box that appears, set **Node Name** and **Description**.
5. Click **OK**.

Configure an output node of the Kafka type

Drag **Kafka** under **Writer** to the canvas. Click the node and then configure the node in the **Node Settings** dialog box that appears.

Parameter	Description
server	The broker server address of Kafka in the format of <code>ip:port</code> .
topic	The name of the Kafka topic.
keyColumn	The column for storing the key.
keyType	The type of the Kafka key.
valueColumn	The column for storing the value. If this parameter is not specified, all columns are concatenated by using the delimiter specified by fieldDelimiter to form the value.
valueType	The type of the Kafka value.
batchSize	The amount of data written at a time. Default value: 1024.

Parameter	Description
Parameters	The extended parameters specified when KafkaConsumer is created, such as bootstrap.servers, auto.commit.interval.ms, and session.timeout.ms. By setting parameters in kafkaConfig, you can control the data consumption behaviors of KafkaConsumer.

Take the following Kafka configuration script as an example.

```
"parameter": {
  "server": "100.81.127.26:9092",
  "keyIndex": 0,
  "valueIndex": 1,
  "keyType": "long",
  "valueType": "string",
  "topic": "xin_002",
  "batchSize": 1,
}
```

2.4.6.6. Transform

2.4.6.6.1. Data filter

The data filter plug-in is used to filter data.

Parameters

Parameter	Description	Required
category	The category of the plug-in. Set this parameter to filter for the data filter plug-in.	Yes
stepType	The type of the step. Set this parameter to filter rows for the data filter plug-in.	Yes
name	The name of the step.	Yes
condition	The filter condition in the expression format. The expression can contain the columns of the reader. In this example, the id and age columns in the filter condition are two columns specified in the reader. An expression that does not contain any column names can also be valid, such as 1==1.	Yes

Example

```
{
  "connections": [],
  "transform": {
    "name": "Data Filter",
    "category": "Data Filter",
    "stepType": "Data Filter",
    "name": "Data Filter",
    "condition": "id > 100"
  }
}
```

```
name : yj_mysql_2_mysql ,
"order":{
  "hops":[
    {
      "from": "J3Bji***NPLn",
      "to": "rYCN***VxmIK0"
    },
    {
      "from": "rYCNF***mIK0",
      "to": "nD***JWPh0YJI0X"
    }
  ]
},
"params": [],
"setting": {
  "keyVersion": "201412091312",
  "metric": {
    "transMetric": {
      ...
    }
  },
},
"steps": [
  {
    "category": "reader",
    "name": "J3BjiG***QNPLn",
    "parameter": {
      "password": "<yourPassWord>",
      "access": "Native",
      "column": [
        "id",
        "name",
        "age"
      ],
      "dbHost": "10.101.83.3",
      "dbName": "test",
      "port": 3306,
      "startTimestampMills": 1551801600000,
      "table": "streamx_test_real_time",
      "type": "MYSQLBINLOG",
      "username": "<yourUserName>"
    },
    "stepType": "mysqlbinlog"
  },
  {
    "category": "filter",
    "name": "rYCNFINr38VxmIK0",
    "parameter": {
      "condition": "(id > 10) &&(age > 30)",
    },
    "stepType": "filterrows"
  },
  ... // The writer configuration is omitted here.
]
```

2.4.6.6.2. String replacement

The string replacement plug-in is used to replace the field values of the String type.

Parameter	Description
Field	The input fields read in the previous step.
Regular Expression Match	Specifies whether to perform regular expression match on the fields. For more information about regular expressions, see the relevant Java documentation.
Original String	The string to be replaced.
New String	The string used to replace the original string.
Case Sensitive	Specifies whether the query of the original string is case-sensitive.

2.4.7. Data synchronization solutions

2.4.7.1. Go to the Sync Solutions page

The Data Integration service of DataWorks allows you to create and configure a synchronization solution to synchronize data from a source to a destination in real time. You can use a synchronization solution to synchronize multiple tables at a time or synchronize both full and incremental data. If you want to synchronize both full and incremental data, you can synchronize the incremental data after the full data is synchronized.

Procedure

1. [Log on to the DataWorks console.](#)
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration.**
3. In the left-side navigation pane, click **Sync Solutions.** The **Solution Task list** page appears. You can create synchronization solutions and view the details and statuses of the created synchronization solutions on this page. A synchronization solution has the following states:
 - **Not running:** The synchronization solution is not run. You can click **Start execution** in the Operation column that corresponds to the synchronization solution to run the synchronization solution.

 **Note** You can click **Task configuration** in the Operation column that corresponds to a synchronization solution in the **Not running** state to edit the synchronization solution. If you click **Task configuration** in the Operation column that corresponds to a synchronization solution in another state, you can only view the information about the synchronization solution.

- **Running:** The synchronization solution is running and cannot be terminated. You must wait until the synchronization solution is completed.
- **Exception:** An error occurred during the running of the synchronization solution. You can click **Execution details** in the Operation column that corresponds to the synchronization solution to troubleshoot the error.
- **Success:** The synchronization solution is completed. You can click **Execution details** in the Operation

column that corresponds to the synchronization solution to view the running results of the synchronization solution.

2.4.7.2. Synchronize data to Hologres in real time

You can create and configure a data synchronization solution to synchronize data in a specified data source to Hologres in real time. This topic describes how to synchronize data to Hologres in real time.

Procedure

1. Go to the **Solution Task list** page.
 - i. [Log on to the DataWorks console](#).
 - ii. Click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. In the left-side navigation pane, click **Sync Solutions**.
2. On the **Solution Task list** page, click **New Task** in the upper-right corner.
3. In the **New resolution task** dialog box, click **One-click real-time synchronization to Hologres**.
4. In the **Set synchronization sources and rules** step, configure the parameters.
 - i. In the **Basic configuration** section, configure the relevant parameters.

Parameter	Description
Scheme name	The name of the synchronization solution. The name can be a maximum of 50 characters in length.
Description	The description of the synchronization solution. The description can be a maximum of 50 characters in length.
Destination task storage location	<p>If you select Automatically establish workflow, DataWorks automatically creates a workflow named in the format of <code>clone_database_Source name+to+Destination name</code> in the Data Integration directory. All synchronization nodes generated by the synchronization solution are placed in the directory of this workflow.</p> <p>If you do not select Automatically establish workflow, you must select a directory from the Select Location drop-down list. All synchronization nodes generated by the synchronization solution are placed in the specified directory.</p>

- ii. In the **Data source** section, specify **Type** and **Data source**.
- iii. In the **Select the source table for synchronization** section, select the tables that you want to synchronize in the **SOURCE Table** list and click **>** to move the tables to the **Selected Source table** list. The **SOURCE Table** list displays all the tables in the source. You can select all or some tables to synchronize them at a time.

 **Notice** If a selected table does not have a primary key, the table cannot be synchronized in real time.

- iv. In the **Set synchronization rules** section, click **Add rule** and select an option to configure naming rules for destination tables. Supported options include **Table name conversion rules** and **Target table name rule**.
 - **Table name conversion rules**: the rule for converting the names of source tables to those of destination tables.
 - **Target table name rule**: the rule for adding a prefix and suffix to the converted names of destination tables.
 - v. Click **Next Step**.
5. In the **Set target table** step, configure the parameters.
- i. Specify **Target Hologres data source** and **Schema**. **Write Hologres policy** is set to **Replay (replay operation log to restore data)** by default and cannot be changed.
 - ii. Click **Reload source table and Hologres Table mapping** to configure the mappings between the source tables and destination Hologres tables.
 - iii. View the mapping progress, source tables, and mapped destination tables. You can view the mapping progress between the source tables and destination tables. The mapping may take a long period of time if you want to synchronize a large number of tables.
- An error message appears if the selected source table does not have a primary key. The synchronization can be performed if one of the selected source tables has a primary key. Source tables without primary keys are ignored during the synchronization.
- You can set **Table creation method** to **Create tables automatically** or **Use existing Table**. The name of the destination table that appears in the **Hologres Table name** column varies based on the setting of **Table creation method**.
- If you set **Table creation method** to **Create tables automatically**, the name of the destination table that is automatically created appears. You can click the table name to view and modify the table creation statements.
 - If you set **Table creation method** to **Use existing Table**, you must select a table from the drop-down list in the **Hologres Table name** column.
- iv. Click **Next Step**.

6. In the **Run resource settings** step, configure the parameters.

Parameter	Description
Realtime task resource group	The resource group used for running the batch synchronization node and real-time synchronization nodes generated by the synchronization solution.
Offline task resource group	
Select scheduling Resource Group	The resource group for scheduling used for running the nodes generated by the synchronization solution.
Maximum number of connections supported by source read	The maximum number of Java Database Connectivity (JDBC) connections that are allowed for the source. Specify an appropriate number based on the resources of the source.
Offline task name rules	The name of the batch synchronization node that is used to synchronize the full data of the source. After a synchronization solution is created, a batch synchronization node is generated first to synchronize full data, and then real-time synchronization nodes are generated to synchronize incremental data.

7. Click **Complete configuration**.

8. On the **Solution Task list** page, find the newly created synchronization solution and click **Start execution** in the Operation column. After the running of the synchronization solution is successful, you can perform the following operations on the synchronization solution:
 - Click **Task configuration** in the Operation column to view the information about or edit the synchronization solution.

 **Note** You can click **Task configuration** in the Operation column that corresponds to the data synchronization solution in the **Not running** state to edit the data synchronization solution. If you click **Task configuration** in the Operation column of a synchronization solution in another state, you can only view information about the synchronization solution.

- Click **Execution details** in the Operation column to view the points in time at which the synchronization solution was started and ended and the status of each node.
- Click **Delete** in the Operation column to delete the synchronization solution. In the **Delete** message, click **Confirm**.

 **Note** After you click **Confirm**, only the configuration record of the data synchronization solution is deleted. The generated synchronization nodes and tables are not affected.

2.4.7.3. Synchronize data to MaxCompute in real time

You can create a synchronization solution to synchronize data from a specified data source to MaxCompute in real time.

Procedure

1. Go to the **Solution Task list** page.
 - i. [Log on to the DataWorks console](#).
 - ii. Click the  icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. In the left-side navigation pane, click **Sync Solutions**.
2. On the **Solution Task list** page, click **New Task** in the upper-right corner.
3. In the **New resolution task** dialog box, click **One-click real-time synchronization to MaxCompute**.
4. In the **Set synchronization sources and rules** step, configure the parameters.

i. In the **Basic configuration** section, configure the relevant parameters.

Parameter	Description
Scheme name	The name of the synchronization solution. The name can be a maximum of 50 characters in length.
Description	The description of the synchronization solution. The description can be a maximum of 50 characters in length.
Destination task storage location	<p>If you select Automatically establish workflow, DataWorks automatically creates a workflow named in the format of <code>clone_database_Source name+to+Destination name</code> in the Data Integration directory. All synchronization nodes generated by the synchronization solution are placed in the directory of this workflow.</p> <p>If you do not select Automatically establish workflow, you must select a directory from the Select Location drop-down list. All synchronization nodes generated by the synchronization solution are placed in the specified directory.</p>

ii. In the **Data source** section, specify **Type** and **Data source**.

iii. In the **Select the source table for synchronization** section, select the tables that you want to synchronize in the **SOURCE Table** list and click **>** to move the tables to the **Selected Source table** list. The **SOURCE Table** list displays all the tables in the source. You can select all or some tables to synchronize them at a time.

 **Notice** If a selected table does not have a primary key, the table cannot be synchronized in real time.

iv. In the **Set synchronization rules** section, click **Add rule** and select an option to configure naming rules for destination tables. Supported options include **Table name conversion rules** and **Target table name rule**.

- **Table name conversion rules:** the rule for converting the names of source tables to those of destination tables.
- **Target table name rule:** the rule for adding a prefix and suffix to the converted names of destination tables.

v. Click **Next Step**.

5. In the **Set target table** step, configure the parameters.

- i. Select a data source from the **Target MaxCompute data source** drop-down list and specify **Write mode**.
- ii. Click  next to **MaxCompute time automatic partition settings**. In the **Edit** dialog box, modify the partition settings for the destination tables. You can configure daily and hourly partitions.
- iii. Click **Reload source table and MaxCompute Table mapping** to configure the mappings between the source tables and destination MaxCompute tables.

- iv. View the mapping progress, source tables, and mapped destination tables. You can view the mapping progress between the source tables and destination tables. The mapping may take a long period of time if you want to synchronize a large number of tables.

An error message appears if the selected source table does not have a primary key. The synchronization can be performed if one of the selected source tables has a primary key. Source tables without primary keys are ignored during the synchronization.

You can set Table creation method to **Create tables automatically** or **Use existing Table**. The name of the destination table that appears in the MaxCompute Table name column varies based on the setting of Table creation method.

- If you set **Table creation method** to **Create tables automatically**, the name of the destination table that is automatically created appears. You can click the table name to view and modify the table creation statements.
- If you set **Table creation method** to **Use existing Table**, you must select a table name from the drop-down list in the MaxCompute Table name column.

v.

6. In the **Run resource settings** step, configure the parameters.

Parameter	Description
Synchronization engine	Default value: Default embedded engine .
Realtime task resource group	The resource group that is used to run the nodes generated by the real-time synchronization solution.
Real-time synchronization task name	The name of the real-time synchronization solution.
Select scheduling Resource Group	The resource group that is used to run the real-time synchronization nodes and batch synchronization node generated by the synchronization solution. Synchronization solutions can run on shared resource groups and custom resource groups for Data Integration.
Offline task resource group	
Maximum number of connections supported by source read	The maximum number of Java Database Connectivity (JDBC) connections that are allowed for the source database. Specify an appropriate number based on the resources of the source database.
Offline task name rules	The name of the batch synchronization node that is used to synchronize the full data of the source database. After a data synchronization solution is created, DataWorks first generates a batch synchronization node to synchronize full data, and then generates real-time synchronization nodes to synchronize incremental data.

7. Click **Complete configuration**.
8. On the **Solution Task list** page, find the newly created synchronization solution and click **Start execution** in the Operation column. After the synchronization solution is run, you can perform the following operations on the synchronization solution:
 - Click **Task configuration** in the Operation column to view information about or configure the synchronization solution.

Note You can configure a synchronization solution only when it is in the **Not running** state. If you click Task configuration in the Operation column of a synchronization solution that is in another state, you can only view information about the synchronization solution.

- Click **Execution details** in the Operation column to view the time at which the synchronization solution was started and ended and the status of each node.
- Click **Delete** in the Operation column to delete the synchronization solution. In the **Delete** message, click **Confirm**.

Note After you click Confirm, only the configuration record of the data synchronization solution is deleted. The generated synchronization nodes and tables are not affected.

2.4.8. Resource groups

2.4.8.1. Overview

This topic introduces the definition of resource groups. It also describes how the connectivity and performance of resource groups affect data synchronization.

Definition

A resource group is a collection of computing resources on which synchronization nodes of Data Integration are run. In most cases, a resource group is one or more servers that consist of CPU, memory, and network resources.

In the process of running a synchronization node, the resource group pulls data from the source and pushes the data to the destination.



Connectivity and performance

When you use resource groups, you must pay attention to their connectivity and performance.

- Connectivity

To ensure that data can be synchronized, a resource group must be connected to the source and destination. Connectivity is the most important factor that affects data synchronization.

Data Integration cannot build networks. Before you use Data Integration to synchronize data, you must make sure that the resource group is connected to the data sources. If the resource group is disconnected from the data sources, synchronization nodes cannot be run.

- Performance

Synchronization nodes consume the CPU, memory, and network resources on the servers where the nodes are run. Insufficient resources may lead to various issues. For example, the nodes fail to be started, wait for resources for a long period of time after startup, transmit data at a low rate, or fail to generate data. To ensure the smooth running of synchronization nodes, you must allocate adequate resources for them.

2.4.8.2. Shared resource groups

Data Integration of DataWorks provides shared resource groups for you to create and run synchronization nodes.

Shared resource groups for Data Integration are created and maintained by Data Integration. Shared resource groups compose a public resource pool. Nodes that use resources in the public resource pool may not be run as scheduled due to insufficient resources. We recommend that you prepare sufficient resources to ensure the efficient running of synchronization nodes.

Note

- You can run a maximum of 25 parallel nodes on a shared resource group for Data Integration when the shared resource group is not in use.
- You cannot change the memory size of a shared resource group. Instead, you can change the number of parallel nodes that can be run on the shared resource group.

The following formula is used to calculate the memory size: $\text{Memory size} = \text{Number of parallel nodes} \times 512 \text{ MB}$.

2.4.9. Full-database migration

2.4.9.1. Overview

This section describes the full-database migration feature in terms of its functions and limits.

Full-database migration is an easy-to-use tool that helps you to improve cost-efficiency. It can quickly upload all the tables in a MySQL database to MaxCompute at a time, saving time that is spent on creating batch tasks for initial data migration to the cloud.

For example, if a database contains 100 tables, you must configure 100 data synchronization tasks in a traditional way. With the full-database migration, you can upload all the tables at a time. However, an upload failure might occur due to the issues that involve the principles of designing database tables.

Task generation rules

After the configuration is completed, MaxCompute tables are created and data synchronization tasks are generated based on the selected tables to be synchronized.

The table names, field names, and field types of the MaxCompute tables are generated according to the advanced settings. If no advanced settings are configured, the structure of MaxCompute tables is identical to that of MySQL tables. The partition of these tables is pt, and its format is yyyyymmdd.

The generated data synchronization tasks are daily scheduled tasks and run automatically on the early morning of the next day. The typical transmission rate is 1 Mbit/s, but it varies depending on the synchronization method and concurrency configurations. **To customize a data synchronization task, locate the task by choosing clone_database > Data Source Name > mysql2odps_table name, and then specify its settings.**

 **Note** We recommend that you perform smoke testing on a data synchronization task on the day when it is generated. **To perform smoke testing, choose Administration Center > Task Management > project_etl_start > Upload Database > Data Source Name, find the synchronization task, right-click the task, and then test the task.**

Limits

Full-database migration has the following limits due to the issues that involve the principles of designing database tables.

- Currently, only the full-database migration from a MySQL data source to MaxCompute is supported. We are working on support for full-database migration from a Hadoop or Hive data source to Oracle.
- Only the daily incremental and daily full upload modes are available.

If you want to synchronize historical data at a time, this feature cannot meet your needs. We recommend that:

- You configure daily tasks instead of synchronizing historical data at a time. You trace the historical data with the provided retrospective data import feature. This eliminates the need to run temporary SQL tasks to split data after all the historical data is synchronized.
- To synchronize historical data at a time, configure a task on the task development page and click **Run**. Then, data is converted by using SQL statements. They are both one-time operations.

If your daily incremental upload task uses a special business logic and cannot be identified by a date field, this feature cannot meet your needs. We provide the following suggestions:

- The incremental data upload can be achieved by using two methods: binlog provided by the DTS product and the date field for data changes provided by databases.

Currently, Data Integration supports the second method. Therefore, your database must contain the date field for data changes. The system determines whether your data is changed on the same day as the business date by using this field. If yes, all the changed data is synchronized.

- To facilitate the incremental data uploading, we recommend that you include the `gmt_create` and `gmt_modify` fields when creating any database tables. Additionally, you can set the `id` field as the primary key to improve efficiency.
- Full-database migration supports batch upload and full upload modes.

Batch upload is configured with time intervals. Currently, the connection pool protection feature for data sources is not supported, but will be available later.

- To prevent overloads on the database, the full-database migration feature provides the batch upload mode. This mode enables you to upload tables in batches at a specified time interval and prevents compromised service functionality. We provide the following suggestions:
 - If you have master and slave databases, we recommend that you synchronize the data of the slave database.
 - In a batch upload task, each table has a database connection with a maximum transmission rate of 1 Mbit/s. For example, if you run a synchronization task for 100 tables at a time, 100 database connections are established. We recommend that you specify proper concurrency settings based on your business needs.
- If you have special requirements for transmission efficiency, this feature cannot meet your needs. The maximum transmission of each generated task is 1 Mbit/s.
- Only the mapping of all table names, field names, and field types are supported.

During the full-database migration process, MaxCompute tables are created automatically, where the partition field is `pt`, the field type is `string`, and the format is `yyyymmdd`.

 **Note** When you select tables for synchronization, all fields must be synchronized and none of these fields can be edited.

2.4.9.2. Migrate a MySQL database

This topic describes how to migrate a MySQL database to MaxCompute.

The database migration feature improves efficiency and reduces costs. It can quickly upload all tables in a MySQL database to MaxCompute. For more information, see [Overview](#).

Procedure

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page.
3. In the left-side navigation pane, choose **Sync Resources > Connections**. On the page that appears, click **Add Connection**.
4. In the **Add Connection** dialog box that appears, select **MySQL**.
5. Add a MySQL connection named `clone_database` for database migration.
6. Click **Test Connection** and verify that the database can be accessed. Click **Complete**.
7. The added MySQL connection named `clone_database` appears in the connection list. Find the added connection and click **Migrate Database** in the Actions column.

The database migration settings page consists of three functional modules.

Functional module	Description
Tables to migrate	This module lists all the tables in the MySQL connection named <code>clone_database</code> . Selected tables will be migrated.
Advanced Settings	You can configure the rules for converting the table name, column names, and data types.
Basic settings	You can select whether to synchronize full or incremental data, whether to upload data in one or more batches, and the synchronization efficiency. You can also view the migration progress and results.

8. Click **Advanced Settings** and configure conversion rules based on your needs. For example, you can add an `ods_` prefix to the name of each MaxCompute table.
9. Specify basic settings. Set Sync Method to Synchronize Incremental Data Daily, and configure the incremental data to be determined based on the `gmt_modified` column. Data Integration will generate WHERE clauses based on the specified column and DataWorks scheduling parameters such as `#{bdp.system.bizdate}`.

Data Integration reads data from MySQL tables by connecting to a remote MySQL database over JDBC and running SELECT statements. Data Integration uses standard SQL statements, and therefore you can configure WHERE clauses to filter data. The WHERE clause used in this example is provided as follows:

```
STR_TO_DATE("#{bdp.system.bizdate}', '%Y%m%d') <= gmt_modified AND gmt_modified < DATE_ADD(STR_TO_DATE("#{bdp.system.bizdate}', '%Y%m%d'), interval 1 day)
```

Select data upload in batches to protect the MySQL database from being overloaded. Let Data Integration start data synchronization for three tables every one hour from 00:00 each day.

Click **Commit**. Then, you can view the migration progress and results of each table.

10. Find table `a1` and click **View Node** to view the migration results.

You have configured a node for migrating a MySQL connection named clone_database to MaxCompute. This node is run based on the specified schedule, daily by default. You can also create retroactive node instances to transmit historical data. The database migration feature of **Data Integration** significantly simplifies the initial configurations for migrating your data to the cloud and reduces data migration costs.

You can view the migration success logs of table a1.

2.4.9.3. Migrate Oracle databases

This topic describes how to migrate an Oracle database to MaxCompute.

The database migration feature improves efficiency and reduces costs. It can quickly upload all tables in an Oracle database to MaxCompute. For more information, see [Overview](#).

Procedure

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page.
3. In the left-side navigation pane, choose **Sync Resources > Connections**. On the page that appears, click **Add Connection** in the upper-right corner.
4. In the **Add Connection** dialog box that appears, select **Oracle**.
5. Add an Oracle connection named clone_database for database migration.
6. Click **Test Connection** and verify that the database can be accessed. Click **Complete**.
7. The added Oracle connection named clone_database appears in the connection list. Find the added connection and click **Migrate Database** in the Actions column.

The database migration settings page consists of three functional modules.

Functional module	Description
Tables to migrate	This module lists all the tables in the Oracle connection named clone_database. Selected tables will be migrated.
Advanced Settings	You can configure the rules for converting the table name, column names, and data types.
Basic settings	You can select whether to synchronize full or incremental data, whether to upload data in one or more batches, and the synchronization efficiency. You can also view the migration progress and results.

8. Click **Advanced Settings** and configure conversion rules based on your needs.
9. Set Sync Method to Synchronize All Data Daily.

 **Note** If a date column exists in your table, you can select incremental migration and configure the incremental data to be determined based on the date column. Data Integration will generate WHERE clauses based on the specified column and DataWorks scheduling parameters such as \${bdp.system.bizdate}.

Select data upload in batches to protect the Oracle database from being overloaded. Let Data Integration start data synchronization for three tables every one hour from 00:00 each day.

Click **Commit**. Then, you can view the migration progress and results of each table.

10. Find a related table and click **View Node** to view the node details.

You have configured a node for migrating an Oracle connection named clone_databae to MaxCompute. This node is run based on the specified schedule, daily by default. You can also create retroactive node instances to transmit historical data. The database migration feature of **Data Integration** significantly simplifies the initial configurations for migrating your data to the cloud and reduces data migration costs.

2.5. Data Analytics

2.5.1. Solution

The data analytics mode of DataWorks is upgraded so that you can group multiple workflows in a solution of a workspace.

Overview

DataWorks upgrades the data analytics mode to organize various types of nodes based on the business category. You can organize workflows to analyze data by business.

By using the data analytics mode that involves the **workspace**, **solution**, and **workflow**, DataWorks defines a new development process and improves user experience.

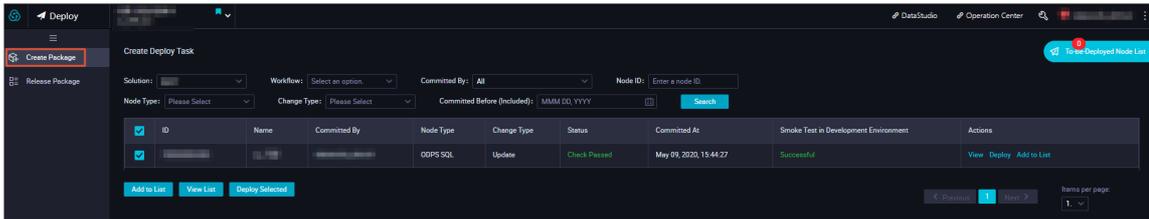
- A workspace is the basic organizational unit that manages the development and O&M permissions of users. The code of all nodes in a workspace can be collaboratively developed and managed by workspace members.
- A solution contains one or more workflows. It has the following advantages:
 - A solution can contain multiple workflows.
 - A workflow can be added to multiple solutions.
 - All solutions in a workspace can be collaboratively developed and managed by workspace members.
- A workflow is an abstract entity of business that enables you to develop data analytics code from a business perspective. A workflow can be added to multiple solutions. It has the following advantages:
 - Workflows facilitate business-oriented code development. Nodes in a workflow are organized by type. A hierarchical directory structure is supported. We recommend that you create a maximum of four levels of sub-directories. To create a sub-directory, right-click the target node type and select **Create Folder**.
 - You can view and optimize each workflow from a business perspective.
 - You can view each workflow on a dashboard to develop code with improved efficiency.
 - You can deploy and manage each workflow as a whole.

Develop a solution

If you double-click a solution in the left-side navigation pane, the left-side navigation pane only displays workflows in the solution. This prevents the development process from being affected by the code that is not related to the current solution in the workspace. To develop a solution, perform the following steps:

1. Log on to the DataWorks console.
2. On the **Data Analytics** tab, move the pointer over  and select **Solution**.
3. In the **Create Solution** dialog box, set **Solution Name** and **Description**, select a workflow from the **Workflows** drop-down list, and then click **Create**.
4. In the solution list, right-click the created solution and select **Solution Kanban**. On the solution dashboard that appears, you can view the selected workflows or modify the solution.
5. Move the pointer over the solution name. The  and  icons appear.

- Click the  icon. The **Deploy** page appears. You can view the nodes to be deployed in the current solution.



 **Note** This icon is available only when the workspace is in standard mode.

- Click the  icon to go to the **Cycle Instance** page under **Cycle Task Maintenance** in **Operation Center**. You can view recurring instances of all nodes in the current solution.

In the left-side navigation pane, double-click the created solution. All the created workflows in the solution appear. You can click a workflow name to show the created nodes in it and perform operations on the nodes and the workflow.

A workflow can be added to multiple solutions. After you develop a solution and add a workflow to the solution, other users can edit the workflow you referenced in their solutions for collaborative development.

2.5.2. SQL coding guidelines and specifications

This topic describes the basic guidelines and detailed specifications of SQL coding.

SQL coding guidelines

The SQL coding guidelines are as follows:

- Make sure that the code is comprehensive.
- Make sure that code lines are clear, neat, well-organized, and structured.
- Consider the optimal execution speed during SQL coding.
- Provide comments whenever necessary to enhance the readability of your code.
- The guidelines impose non-mandatory constraints on the coding behavior of developers. In practice, understandable deviations are allowed when developers obey general rules.
- Use lowercase letters for all keywords and reserved words. Keywords and reserved words include select, from, where, and, or, union, insert, delete, group, having, and count.
- In addition to keywords and reserved words, other code such as field names and table alias must be in lowercase.
- A unit of indentation contains four spaces. All indentations must be the integral multiple of an indentation unit. The code is aligned according to its hierarchy.
- The `select *` operation is prohibited. The column name must be specified for all operations.
- Matching opening and closing parentheses must be placed in the same column.

SQL coding specifications

The SQL coding specifications are as follows:

- Code header

The code header contains information such as the subject, description, author, and date. Reserve a line for change log and a title line so that later users can add change records. Each line can contain a maximum of 80 characters. The template is as follows:

```
-- MaxCompute(ODPS) SQL
_*****
-- ** Subject: Transaction
-- ** Description: Transaction refund analysis
-- ** Author: Youma
-- ** Created on: 20170616
-- ** Change log:
-- ** Modified on Modified by Content
-- yyyyymmdd name comment
-- 20170831 Wuma Add a comment on the biz_type=1234 transaction
_*****
```

- Field arrangement
 - Use a line for each field that is selected for the SELECT statement.
 - Separate the first field from SELECT by one indentation unit.
 - Enter another field name in a separate line after two indentation units.
 - Place the comma (,) between two fields right before the second field.
 - Place the AS statement in the same line as the corresponding field. We recommend that you keep the AS statements of multiple fields in the same column.

```
select  channel_id          as channel_id
        , trade_channel_desc as trade_channel_desc
        , trade_channel_edesc as trade_channel_edesc
        , inst_date         as inst_date
        , trade_iswap       as trade_iswap
        , channel_type      as channel_type
        , channel_second_desc as channel_second_desc
from    (
```

- Clause arrangement for an INSERT statement

Arrange the clauses of an INSERT statement in the same line.
- Clause arrangement for a SELECT statement

The clauses such as FROM, WHERE, GROUP BY, HAVING, ORDER BY, JOIN, and UNION in a SELECT statement must be arranged according to the following requirements:

 - Use a line for each clause.
 - Make sure that the clauses are left aligned with the SELECT statement.
 - Add two indentation units between the first word and the other code in a clause.
 - Keep the logical operators such as AND and OR in a WHERE clause left aligned with WHERE.

- If the length of a clause name exceeds two indentation units such as ORDER BY and GROUP BY, add a space between the clause name and its content.

```
select      trim(channel) channel
            ,min(id)      id
from        ods_trd_trade_base_dd
where       channel is not null
and         dt = ${tmp_uuuummdd}
and         trim(channel) <> ''
group by    trim(channel)
order by    trim(channel)
```

- Spacing before and after operators

Keep one space before and one space after the arithmetic and logical operators and keep the operators in the same line, unless the clause contains more than 80 characters.

```
select      trim(channel) channel
            ,min(id)      id
from        ods_trd_trade_base_dd
where       channel is not null
and         dt = ${tmp_uuuummdd}
and         trim(channel) <> ''
group by    trim(channel)
order by    trim(channel)
```

- CASE statement

The CASE statement can be used to determine the value of a field in a SELECT statement. Rules for writing CASE statements are as follows:

- Place the WHEN clause in the same line as the CASE statement, with one indentation unit between them.
- Keep a WHEN clause in one line whenever possible. If the statement is long, line breaks can be made.
- A CASE statement must contain an ELSE clause. The ELSE clause must be aligned with the WHEN clause.

```
, case      when p1.trade_from = '3008' and p1.trade_email is null then 2
            when p1.trade_from = '4000' and p1.trade_email is null then 1
            when p9.trade_from_id is not null then p9.trade_from_id
end
,p1.trade_email as trade_from_id
               as partner_id
```

- Nested query

Nested queries are often used in extract-transform-load (ETL) development of data warehouse systems. The following figure shows an example of a nested query.

```

select      p.channel
           ,rownumber() order_id
from        (
           select  s1.channel
                ,s1.id
           from      (
                   select  trim(channel)      as channel
                        ,min(id)             as id
                   from    ods_trd_trade_base_dd
                   where   channel is not null
                   and     dt = ${tmp_yyyymmdd}
                   and     trim(channel) <> ''
                   group  by trim(channel)
                ) s1
           left outer join
                dim_trade_channel s2
           on    s1.channel = s2.trade_channel_edesc
           where s2.trade_channel_edesc is null
           order by id
        ) p
;

```

- Table alias

- Once an alias is defined for a table in a SELECT statement, use the alias whenever you reference the table in the statement. Therefore, you must specify an alias for each table.
- We recommend that you define the table aliases with simple characters, such as a, b, c, and d in sequence, and avoid using keywords.

- In the nested query, levels 1 to 4 of SQL statements are named part, segment, unit, and detail, which are abbreviated as P, S, U, and D. You can also use a, b, c, and d to represent levels 1 to 4.

To differentiate multiple clauses at the same level, add numbers such as 1, 2, 3, and 4 next to the letters. Add comments to the table aliases as needed.

```

select      p.channel
            ,rownumber() order_id
from        (
            select  s1.channel
                ,s1.id
            from    (
                    select  trim(channel)      as channel
                        ,min(id)              as id
                    from    ods_trd_trade_base_dd
                    where   channel is not null
                    and     dt = ${tmp_yyyymmdd}
                    and     trim(channel) <> ''
                    group by trim(channel)
                ) s1
            left outer join
                dim_trade_channel s2
            on   s1.channel = s2.trade_channel_edesc
            where s2.trade_channel_edesc is null
            order by id
        ) p
;

```

- SQL comments
 - Add a comment for each SQL statement.
 - Use a separate line for the comment of each SQL statement and place the comment in front of the SQL statement.
 - Place the comment of a field right after the field.
 - Add comments for clauses that are difficult to understand.
 - Add comments for important code.
 - If a statement is long, we recommend that you add comments based on the purposes of each segment.
 - The description for a constant or variable is required. The comment on the valid value range is optional.

2.5.3. GUI elements

2.5.3.1. Overview

This topic describes the graphical user interface (GUI) elements on the DataStudio page and the configuration tab of an ODPS SQL node.

Log on to the DataWorks console. The **Data Analytics** page appears. You can double-click a created node to perform operations on the node configuration tab.

The following table describes the GUI elements.

No.	GUI element	Description
1	Show My Nodes Only icon	Click the icon to view your own nodes.

No.	GUI element	Description
2	Search Code icon	Click the icon to search for a node or a code segment.
3	Create icon	Click the icon to create a solution, workflow, folder, node, table, resource, or function.
4	Refresh icon	Click the icon to refresh the directory tree in the left-side navigation pane.
5	Locate icon	Click the icon to find the current node in the left-side navigation pane.
6	Import icon	<p>Click the icon to import local data to an online table. You must specify the encoding format.</p> <div style="background-color: #e0f2f1; padding: 5px; border: 1px solid #ccc;"> <p> Note In a workspace of the standard mode, the local data is imported to a table in the development environment.</p> </div>
7	Filter icon	Click the icon to query nodes based on the specified filter conditions.
8	Save icon	Click the icon to save the code of the current node.
9	Save as Ad-Hoc Query Node icon	Click the icon to save the code of the current node in an ad-hoc query node. You can find the node on the Ad-Hoc Query tab.
10	Commit icon	Click the icon to commit the current node.
11	Commit and Unlock icon	Click the icon to commit and unlock the current node for editing.
12	Steal Lock icon	Click the icon to steal the lock of the current node and then edit it if you are not the owner of the node.
13	Run icon	Click the icon to run the code of the current node. You only need to assign values to variables in SQL statements once. The initial values are retained even if the node code changes.
14	Run with Arguments icon	<p>Click the icon to run the code of the current node with the configured parameters. You must manually assign values to variables in SQL statements each time you click this icon. The initial values are passed to the Run with Arguments feature, which replaces the initial values with the assigned values.</p> <p>For example, if the run date of a node is set to April 2, the node always runs on April 2 when you click the Run icon. After you click Run with Arguments icon and change the run date to April 3, the run date is updated. When you click the Run icon again, the node is run on April 3.</p>
15	Stop icon	Click the icon to stop running the code of the current node.
16	Reload icon	Click the icon to reload the code of the current node. The code will be restored to the version last saved. Unsaved changes will be lost.

No.	GUI element	Description
17	Run Smoke Test icon	<p>Click the icon to test the code of the current node. A smoke test allows you to replace the values of scheduling parameters in the specified data timestamp with your simulated ones. This feature tests the effect of value changes for scheduling parameters.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note Each time after you modify the values of scheduling parameters, you must save and commit the modification before running the smoke test. Otherwise, the new values of scheduling parameters do not take effect.</p> </div>
18	View Smoke Test Log icon	Click the icon to view the runtime logs of the current script template.
19	Format Code icon	Click the icon to format the code to avoid excessively long code in a single line.
20	Operation Center button	Click the icon to go to Operation Center.
21	Properties tab	Click the tab to configure the properties such as the scheduling properties, parameters, and resource group for the current node.
22	Lineage tab	Click the tab to view the relationships between the current node and other nodes.
23	Versions tab	Click the tab to view the committed and deployed versions of the current node.
24	Code Structure tab	Click the tab to view the code structure of the current node. If the code is excessively long, you can quickly find a code segment based on the key information in the structure.

2.5.3.2. Workflow Parameters

On the Workflow Parameters tab, you can assign a value to a variable or replace the value of a parameter for all nodes in the current workflow. This topic describes how to configure a workflow parameter by assuming that you want to replace the value of the ReplaceMe parameter with ReplaceMe123 in a manually triggered workflow.

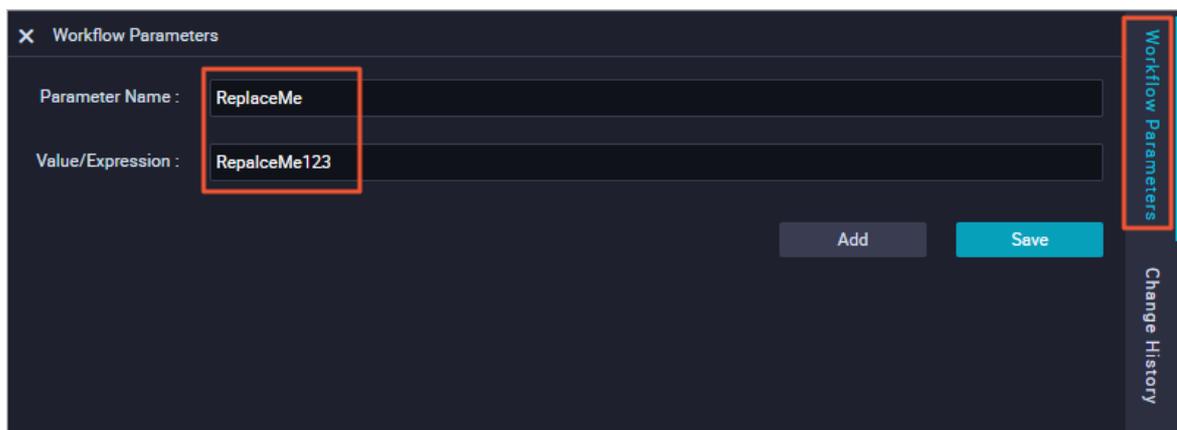
Limits

- In manually triggered workflows, ODPS SQL nodes, Shell nodes, and sync nodes support global parameters. The format for specifying a global parameter varies based on the node type. For example, a global workflow parameter is specified as `x=y1`.
 - To configure the workflow parameter for an ODPS SQL node, double-click the target node and click the **General** tab in the right-side navigation pane. On the **General** tab, enter `x=aaa` in the Arguments field. When the node is run, `x=aaa` specified in the Arguments field is replaced with `x=y1`. You can use `$x` to reference the workflow parameter in the code.
 - To configure the workflow parameter for a Shell node, double-click the target node and click the **General** tab in the right-side navigation pane. On the **General** tab, enter `$x` in the Arguments field. When the node is run, `x=aaa` specified in the Arguments field is replaced with `y1`. You can use `$1` to reference the workflow parameter in the code.

- To configure the workflow parameter for a sync node, double-click the target node and click the **General** tab in the right-side navigation pane. On the **General** tab, enter `-p"-Dx=aaa"` in the Arguments field. When the node is run, `x=aaa` specified in the Arguments field is replaced with `-p"-Dx=y1`. You can use `$x` to reference the workflow parameter in the code.
- In auto triggered workflows, only ODPS SQL nodes support global parameters.
- Parameter names and values are case-sensitive.

Configure a workflow parameter

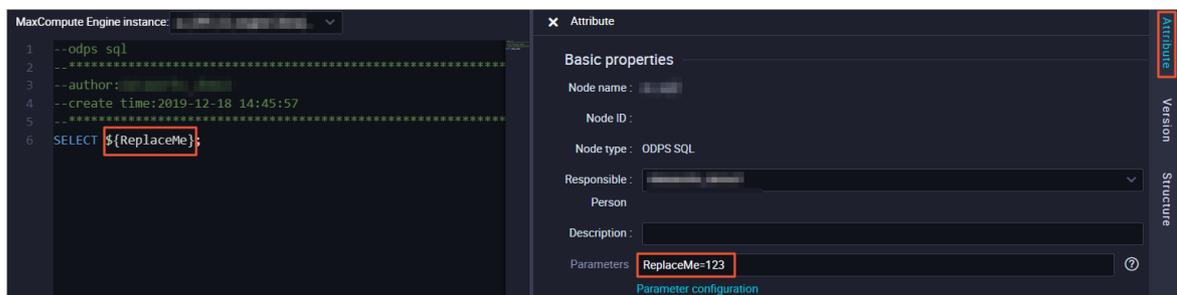
1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **Manually Triggered Workflows**.
3. Double-click the target workflow to go to the workflow configuration tab.
4. In the right-side navigation pane, click the **Workflow Parameters** tab. In the Workflow Parameters pane, enter `ReplaceMe` in the **Parameter Name** field and `ReplaceMe123` in the **Value/Expression** field.



5. Click  in the toolbar.

Configure the workflow parameter for an ODPS SQL node

1. On the DataStudio page, click **Manually Triggered Workflows** in the left-side navigation pane.
2. Find the target workflow and choose **MaxCompute > Data Analytics** to show all the existing data analytics nodes. Double-click the target ODPS SQL node to go to the node configuration tab.
3. In the right-side navigation pane, click the **General** tab. In the General pane, enter `ReplaceMe=123` in the **Arguments** field.

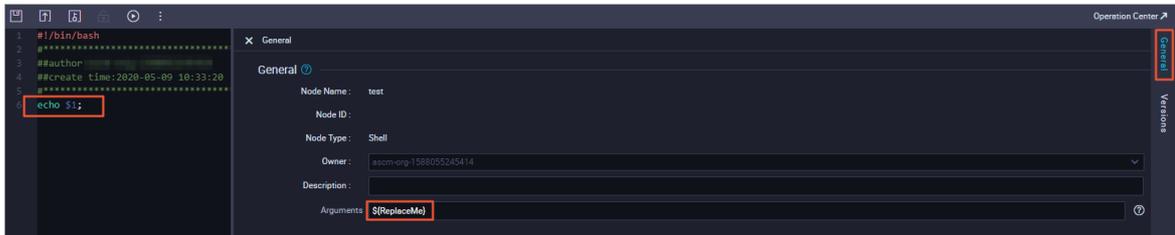


The workflow parameter is specified as `ReplaceMe=ReplaceMe123`. Therefore, the workflow parameter `ReplaceMe` is assigned the value `ReplaceMe123` for this node when the workflow is run.

4. Click  in the toolbar.

Configure the workflow parameter for a Shell node

1. On the **DataStudio** page, click **Manually Triggered Workflows** in the left-side navigation pane.
2. Find the target workflow and click **General** to show all the existing data analytics nodes. Double-click the target Shell node to go to the node configuration tab.
3. In the right-side navigation pane, click the **General** tab. In the General pane, enter `${ReplaceMe}` in the **Arguments** field.



Note Make sure that you enter the parameter in the correct format.

4. Click  in the toolbar.

Configure the workflow parameter for a sync node

1. On the **DataStudio** page, click **Manually Triggered Workflows** in the left-side navigation pane.
2. Find the target workflow and click **Data Integration** to show all the existing data integration nodes. Double-click the target sync node to go to the node configuration tab.
3. In the right-side navigation pane, click the **General** tab. In the General pane, enter `-p"ReplaceMe=abc"` in the **Arguments** field.

Note Make sure that you enter the parameter in the correct format, namely, `-p"-DParameter name=Parameter value"`.

4. Click  in the toolbar.

Run the workflow to view the result

On the configuration tab of the workflow, click  in the toolbar. In the Warning dialog box, click **Settings**. In the **Runtime Parameters** dialog box, set **Arguments** to `ReplaceMe`. The value of the workflow parameter is replaced when the workflow is run.

You can use the following methods to view the value assigned to the workflow parameter for different types of nodes:

- Right-click the ODPS SQL node and select **View Log**. Then, you can view the value assigned to the workflow parameter for the ODPS SQL node.
- Right-click the Shell node and select **View Log**. Then, you can view the value assigned to the workflow parameter for the Shell node.
- Right-click the sync node and select **View Log**. Then, you can view the value assigned to the workflow parameter for the sync node.

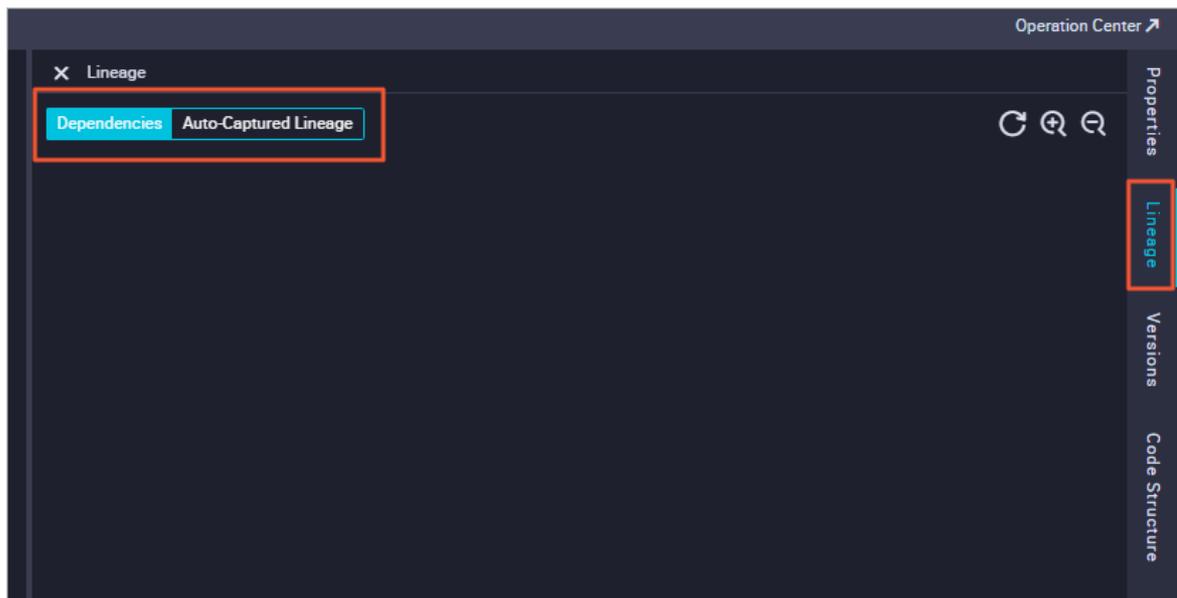
If you have not assigned a value to a workflow parameter on the **Workflow Parameters** tab for a manually triggered workflow, you must assign a value to the workflow parameter every time you run the workflow in the production environment.

2.5.3.3. Lineage

The Lineage tab displays the relationships between a node and other nodes. You can view the node dependencies and the lineage parsed from the code of the node.

Go to the Lineage tab

1. Log on to the DataWorks console.
2. Double-click the target node. For more information about how to create a node, see [Create an ODPS SQL node](#).
3. On the node configuration tab that appears, click the **Lineage** tab in the right-side navigation pane.



On the **Lineage** tab, you can click **Dependencies** to view the dependencies or click **Auto-Captured Lineage** to view the lineage.

View the dependencies

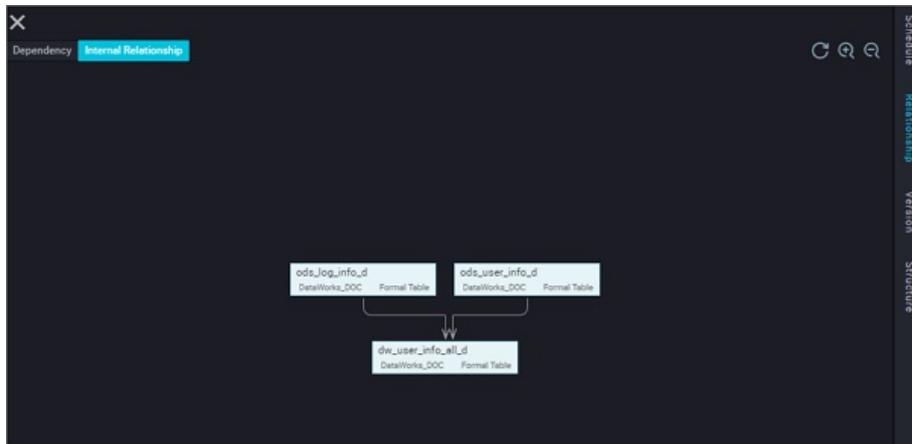
You can check the node dependencies presented based on the current configuration. If the node dependencies fail to meet your expectations, you can reconfigure the node dependencies on the Properties tab.

View the auto-captured lineage

The lineage is parsed based on the code of the current node. For example, an ODPS SQL node contains the following SQL statements:

```
INSERT OVERWRITE TABLE dw_user_info_all_d PARTITION (dt='${bdp.system.bizdate}')
SELECT COALESCE(a.uid, b.uid) AS uid
, b.gender
, b.age_range
, b.zodiac
, a.region
, a.device
, a.identity
, a.method
, a.url
, a.referer
, a.time
FROM (
  SELECT *
  FROM ods_log_info_d
  WHERE dt = ${bdp.system.bizdate}
) a
LEFT OUTER JOIN (
  SELECT *
  FROM ods_user_info_d
  WHERE dt = ${bdp.system.bizdate}
) b
ON a.uid = b.uid;
```

The following figure shows the lineage parsed from the preceding SQL statements. The results queried from the ods_log_info_d and ods_user_info_d tables are joined and then inserted into the dw_user_info_all_d table.



2.5.3.4. Versions

The Versions tab displays all committed and deployed versions of a node. You can view the historical versions and information about each version, including the user who committed the version, time when the version was committed, change type, status, and description.

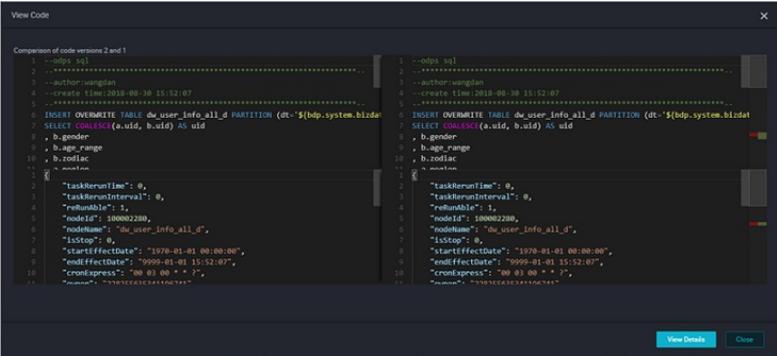
Note Only a committed node has the version information. Every time a node is committed, a version is generated and added to the Versions tab.

1. Log on to the DataWorks console.
2. On the **DataStudio** page, double-click the target node.
3. On the node configuration tab that appears, click the **Versions** tab in the right-side navigation pane. In

the Versions pane, view the committed and deployed versions of the current node.

Module	Relationship	Version	Structure					
<input type="checkbox"/>	5000118 87	V7	dataworks_demo2	2018-09-02 10:3 9:57	Edit	Published	test	View Code Roll Back
<input type="checkbox"/>	5000118 87	V6	dataworks_demo2	2018-09-02 10:3 7:47	Edit	Published	123	View Code Roll Back
<input type="checkbox"/>	5000118 87	V5	dataworks_demo2	2018-09-02 10:3 6:28	Edit	Published	test	View Code Roll Back
<input type="checkbox"/>	5000118 87	V4	dataworks_demo2	2018-09-02 10:3 3:54	Edit	Published	test	View Code Roll Back
<input type="checkbox"/>	5000118 87	V3	dataworks_demo2	2018-09-02 10:3 0:19	Edit	Published	test	View Code Roll Back
<input type="checkbox"/>	5000118 87	V2	wangdan	2018-08-31 10:2 1:19	Edit	Published	workshop user portrait part is written logically.	View Code Roll Back
<input type="checkbox"/>	5000118 87	V1	wangdan	2018-08-30 17:3 7:55	Add	Published	workshop user portrait part is written logically.	View Code Roll Back

GUI element	Description
File ID	The ID of the node.
Versions	The version of the node. A version is generated each time the node is committed and deployed. V1 indicates version 1 and V2 indicates version 2. The version number is incremented by 1 each time.
Committed By	The user who committed the version.
Committed At	The time when the version was committed. If a version is committed and then deployed at a later time point, the value of this parameter is updated to the time when the version is deployed. By default, this column records the time when the version is last operated.
Change Type	The operation on the node. The value of this parameter is Create if the node is committed and deployed for the first time or Change if the node is modified, committed, and then deployed.
Status	The status of the version. Valid values: <ul style="list-style-type: none"> ◦ Yes: The version is committed to the development environment but the related deployment task has not been created. The version has not been deployed in the production environment. ◦ Not Deployed: The version is committed to the development environment and the deployment task is created. The version is pending for deployment. ◦ Deployed: The version is committed to the development environment and deployed in the production environment.
Description	The change description of the version when it is committed. This description helps other users find the relevant version when they manage the node.

GUI element	Description
<p>Actions</p>	<p>The actions that you can perform on the version. Two actions are available: View Code and Roll Back.</p> <ul style="list-style-type: none"> ◦ View Code: Click the button to view the code of the current version. ◦ Roll Back: Click the button to roll back the node from the current version to the required version. After you roll back a node, you must commit and deploy it again.
<p>Compare</p>	<p>Click the button to compare the code and properties between two selected versions.</p>  <p>Click View Details. On the details page that appears, you can view the changes in code and properties.</p> <p>Note You can only compare two versions and cannot compare one or more than two versions at a time.</p>

2.5.3.5. Code Structure

The Code Structure tab displays the SQL code structure parsed from the code of a node. The code structure helps you view and modify the code.

1. Log on to the DataWorks console.
2. Double-click the ODPS SQL node whose code structure you want to view. For more information about how to create a node, see [Create an ODPS SQL node](#).
3. On the node configuration tab that appears, click the **Code Structure** tab in the right-side navigation pane.

```

1  --odps sql
2  .....
3  --author: .....
4  --create time:2020-05-09 11:16:53
5  .....
6  INSERT OVERWRITE TABLE dw_user_info_all_d PARTITION (dt='${bdp.system.bizdate}')
7  SELECT COALESCE(a.uid, b.uid) AS uid
8     , b.gender
9     , b.age_range
10    , b.zodiac
11    , a.region
12    , a.device
13    , a.identity
14    , a.method
15    , a.url
16    , a.referer
17    , a.time
18  FROM (
19     SELECT *
20     FROM ods_log_info_d
21     WHERE dt = '${bdp.system.bizdate}')
22     ) a
23  LEFT OUTER JOIN (
24     SELECT *
25     FROM ods_user_info_d
26     WHERE dt = '${bdp.system.bizdate}')
27     ) b
28  ON a.uid = b.uid;

```

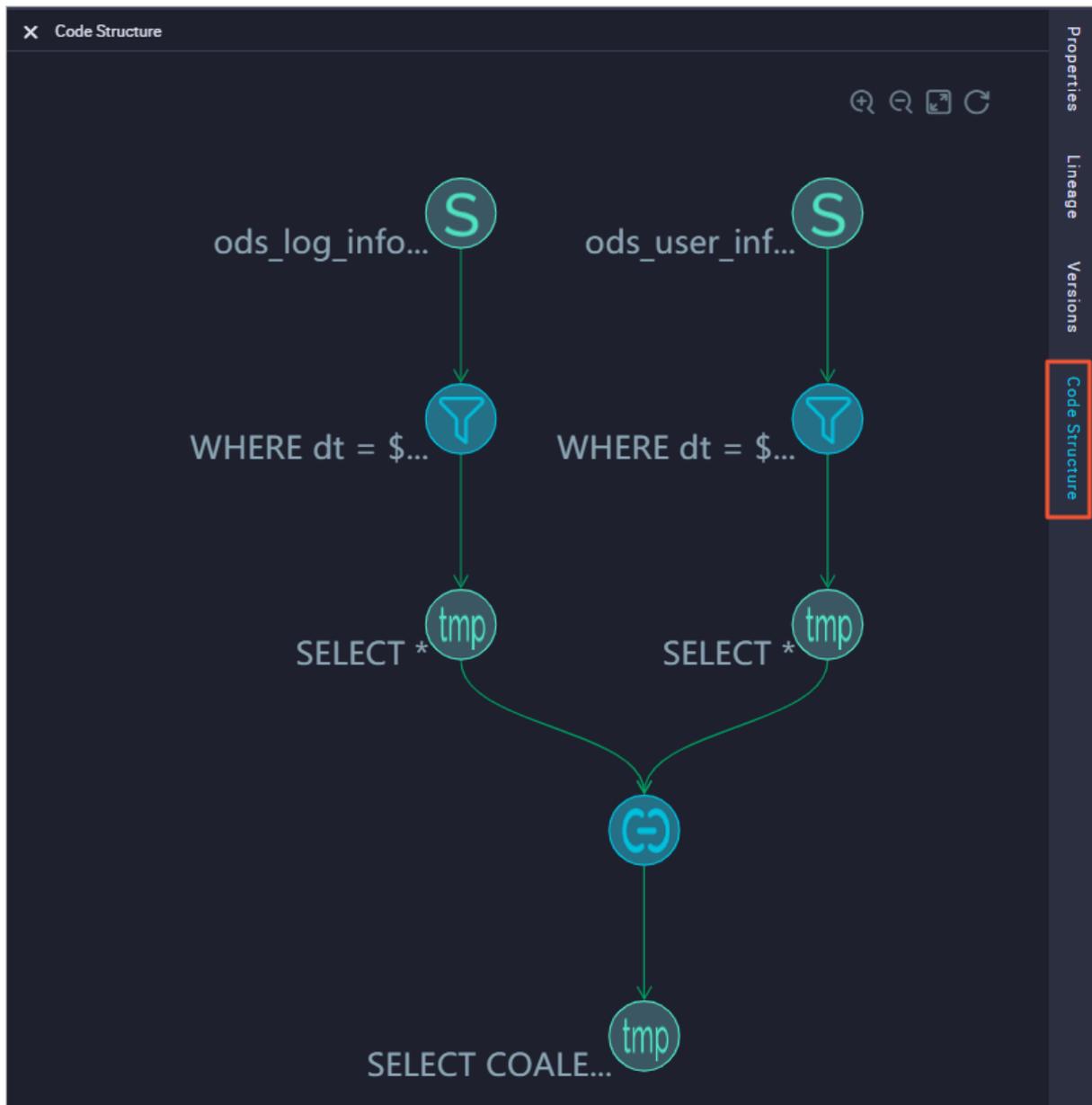
In this example, the ODPs SQL node contains the following SQL statement:

```

INSERT OVERWRITE TABLE dw_user_info_all_d PARTITION (dt='${bdp.system.bizdate}')
SELECT COALESCE(a.uid, b.uid) AS uid
, b.gender
, b.age_range
, b.zodiac
, a.region
, a.device
, a.identity
, a.method
, a.url
, a.referer
, a.time
FROM (
  SELECT *
  FROM ods_log_info_d
  WHERE dt = '${bdp.system.bizdate}')
) a
LEFT OUTER JOIN (
  SELECT *
  FROM ods_user_info_d
  WHERE dt = '${bdp.system.bizdate}')
) b
ON a.uid = b.uid;

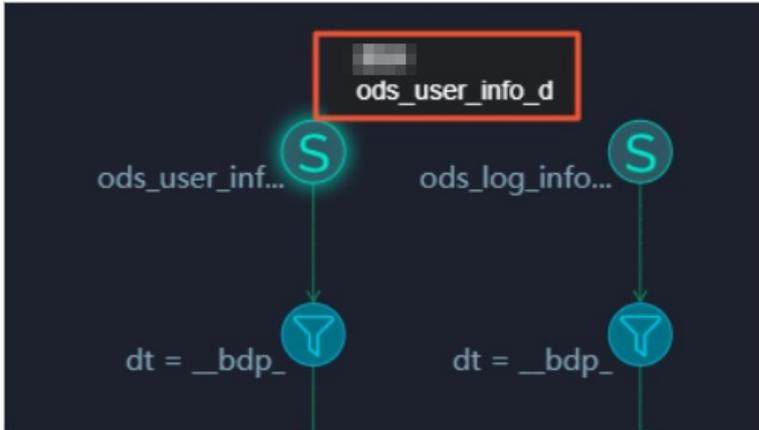
```

The following figure shows the code structure parsed from the preceding SQL statement.

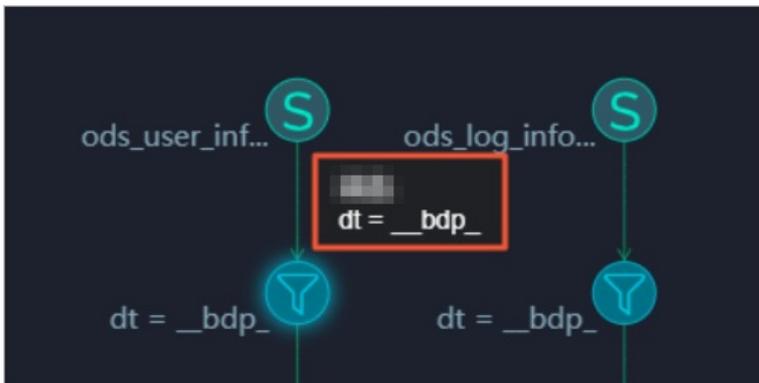


Move the pointer over a circle to view the description.

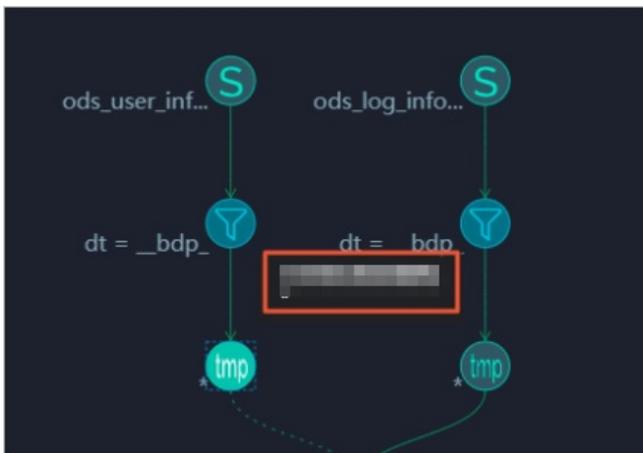
- **Source table:** the table to be queried.



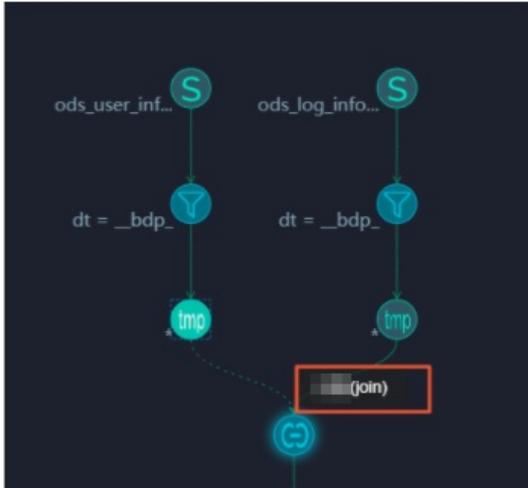
- **Filter:** the condition for filtering the partitions in the table to be queried.



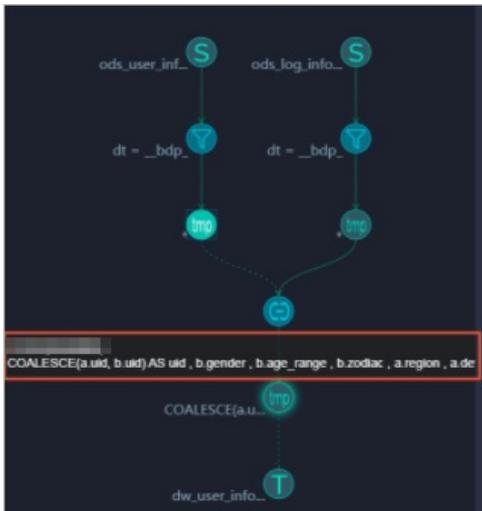
- **First intermediate table (view):** the temporary table that stores the query results.



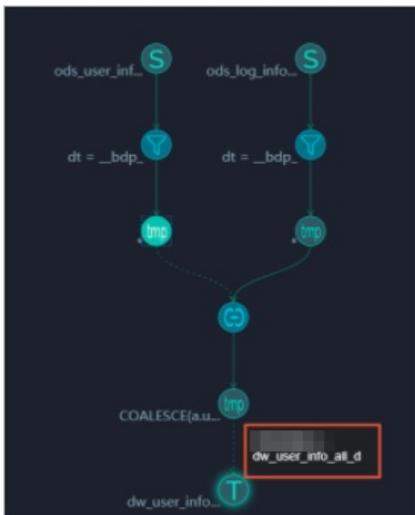
- **Join:** the operation for joining the query results.



- **Second intermediate table (view):** the temporary table that stores the results of the JOIN operation. This temporary table can be stored for three days. After three days, this table is automatically deleted.



- **Destination table (insert):** the destination table to which the query results are inserted by using an **INSERT OVERWRITE** statement.



2.5.4. Business flows

2.5.4.1. Overview

DataWorks organizes different types of nodes in a workflow by business category. This allows you to develop code by business.

DataWorks provides you with a dashboard for different types of nodes in each workflow. DataWorks also provides tools for you to optimize and manage nodes in each workflow. This promotes easy and intelligent development and management.

Workflow structure

A workspace supports multiple types of compute engines and multiple workflows. A workflow is a collection of various types of nodes that are closely associated with each other. DataWorks automatically generates a DAG so that you can view the workflow. A workflow supports the following types of nodes: data integration, data analytics, table, resource, function, and algorithm.

Each type of node has an independent folder. You can also create subfolders in each folder. To facilitate management, we recommend that you create a maximum of four levels of subfolders. If more than four subfolder levels are required, your workflow is too complex. We recommend that you split the workflow into two or more workflows and add the split workflows to one solution.

Create a workflow

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **Data Analytics**.
3. On the **Data Analytics** tab, right-click **Business Flow** and select **Create Workflow**.
4. In the **Create Workflow** dialog box, set **Workflow Name** and **Description**.

 **Notice** The name of the workflow, which cannot exceed 128 characters in length.

5. Click **Create**.

Workflow nodes

A workflow consists of the following types of nodes:

- **Data integration**

Click the target workflow and double-click **Data Integration** to view all data integration nodes of the workflow.

- **MaxCompute**

The MaxCompute engine supports various data analytics nodes, such as ODPS SQL, SQL Snippet, ODPS Spark, PyODPS, ODPS Script, and ODPS MR nodes. You can also view and create tables, resources, and functions.

- **Data analytics**

Right-click **Data Analytics** under **MaxCompute** in the target workflow and select **Create** to create a data analytics node of a specific type.

- **Table**

Click the target workflow and choose **Create > Table** under **MaxCompute** to create a table. You can also view all the tables that are created in the current MaxCompute project.

- **Resource**

Click the target workflow, choose **Create > Resource** under **MaxCompute**, and then click a specific resource type to create a resource. You can also view all the resources that are created in the current MaxCompute project.

- **Function**

Click the target workflow and choose **Create > Function** under **MaxCompute** to create a function. You can also view all the functions that are created in the current MaxCompute project.

- **EMR**

The E-MapReduce compute engine supports the following types of data analytics nodes: EMR Hive, EMR MR, EMR Spark, and EMR Spark SQL. You can also view and create E-MapReduce resources.

Note The EMR folder is available only after you create an E-MapReduce compute engine on the Project Management page.

- **Data analytics**

Click the target workflow, right-click **Data Analytics** under **EMR**, and then select **Create** to create a data analytics node of a specific type.

- **Resource**

Click the target workflow, right-click **Resource** under **EMR**, and then select **Create** to create a resource of a specific type. You can also view all the resources that are created in the current E-MapReduce compute engine.

- **Algorithm**

Click the target workflow, right-click **Algorithm**, and then choose **Create > PAI Experiment** to create a PAI Experiment node. You can also view all the PAI Experiment nodes that are created in the current workflow.

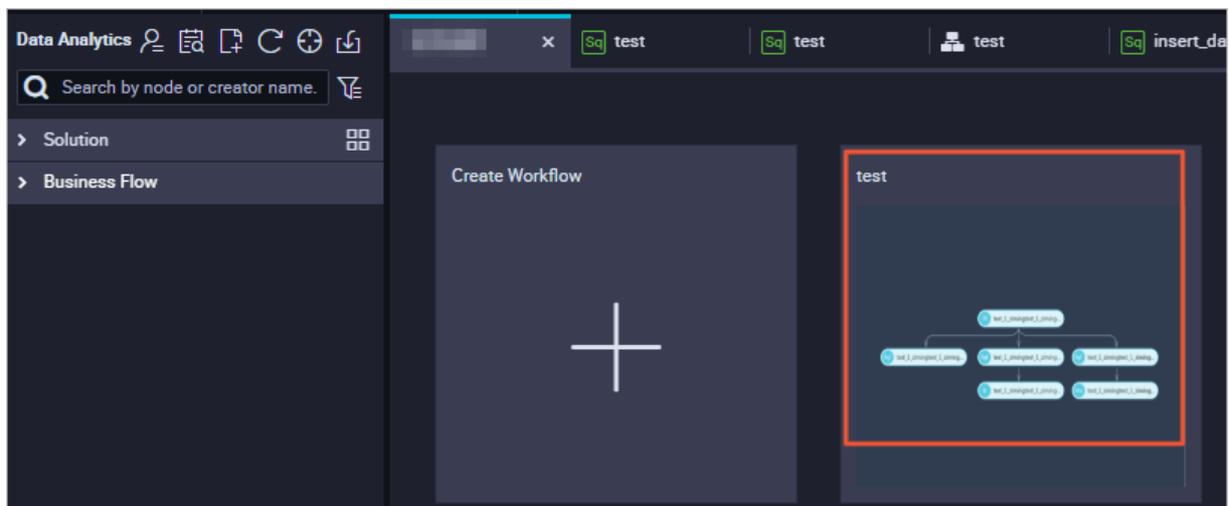
- **General**

Click the target workflow, right-click **General**, and then select **Create** to create a node of a specific type.

View all workflows

On the **Data Analytics** tab, right-click **Business Flow** and select **All Workflows** to view all workflows that are created in the current workspace.

Click a workflow. The dashboard of the workflow appears.



View the dashboard for each node type

DataWorks provides a dashboard for each type of nodes in a workflow. On the dashboard, each node is presented by a card that offers operation and optimization suggestions, so that you can intelligently manage nodes.

For example, the card of each data analytics node provides two indicators to show whether baseline-based monitoring and event notification are enabled for the node. This allows you to understand the status of each node.

You can double-click a folder in a workflow to view the dashboard of the selected node type.

Commit a workflow

1. Go to the dashboard of a workflow and click  in the toolbar.
2. In the **Commit** dialog box, select the nodes to be committed, set **Description**, and then select **Ignore I/O Inconsistency Alerts**.
3. Click **Commit**.

 **Note** If a node has been committed but the node code is not modified, the node cannot be selected again. In this case, you can enter your comments on the node and click **Commit**. The property changes of the node are automatically committed.

2.5.4.2. Create and reference a node group

This topic describes how to create and reference a node group.

Context

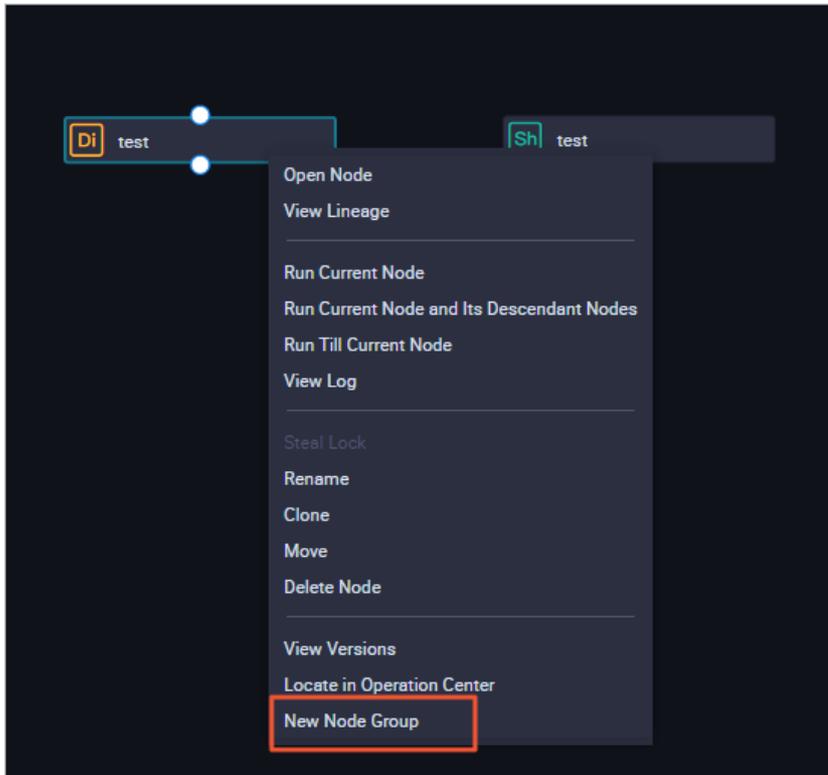
You can group several nodes that are frequently reused together as a node group. The configuration of each node remains unchanged after the nodes are added to a node group. Later, you can directly reference the node group to reuse these nodes.

Create a node group

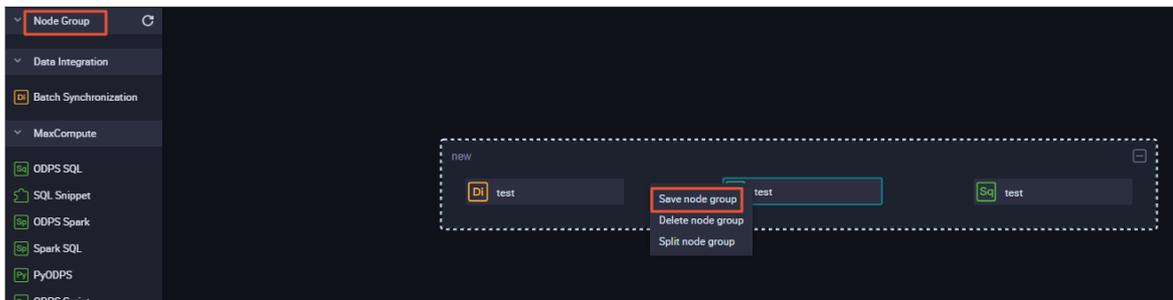
1. Log on to the DataWorks console.
2. On the **Data Analytics** tab, create a workflow. For more information, see [Create a workflow](#).
3. Go to the dashboard of the created workflow. Click  in the upper-right corner and drag a box to select the target nodes to be included in a node group.



4. Right-click any node among the selected nodes and select **New Node Group**.



5. In the **New Node Group** dialog box, enter a name in the **Name** field and click **OK**.
6. Right-click the node group and select **Save node group**. In the dialog box that appears, click **OK**. Then, you can view the created node group in the **Node Group** section.



Menu item	Description
Save node group	Save the node group. The node group that you have created appears in the Node Group section only after you click Save node group . A node group that is not saved cannot be referenced in other workflows.
Delete node group	Delete the node group. Click Delete node group to delete all nodes in the selected node group.
Split node group	Dismiss the node group. After the node group is dismissed, the selected nodes no longer form a node group in the workflow. However, the node group still exists in the Node Group section.

Note If the created node group contains a PAI Experiment node, create a PAI experiment in another workflow to reference the node group. If the created node group contains a branch node, add digits to the value in the **Associated Node Output** parameter.

Reference a node group

You can directly drag a node group to another workflow to reference the node group in the workflow. The dependencies among the nodes in the node group remain unchanged.

You can run the workflow or commit and deploy the workflow. Then, go to **Operation Center** to view the running result.

2.5.5. Node types

2.5.5.1. Data Integration

2.5.5.1.1. Create a batch sync node

Batch sync nodes support various types of data stores, including MaxCompute, MySQL, DRDS, SQL Server, PostgreSQL, Oracle, MongoDB, Db2, Table Store, OSS, FTP, HBase, LogHub, HDFS, and Stream.

Context

When you enter a table name, a drop-down list appears, displaying all matched tables. Only exact match is supported. Therefore, you must enter a complete table name. Tables are labeled as unsupported if they are not supported by batch sync nodes.

If you move the pointer over a table in the list, the details of the table appear, including the database, IP address, and owner of the table. After you select a table, the column information is automatically entered. You can add, move, and delete columns.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **Data Integration > Batch Synchronization**. Alternatively, you can click a workflow in the Business Flow section, right-click **Data Integration**, and then choose **Create > Batch Synchronization**.
3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.). It is not case-sensitive.

4. Click **Commit**.
5. Configure the batch sync node. For more information, see [Overview](#).
6. On the configuration tab of the batch sync node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, set parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.

 **Notice** You must set **Rerun** and **Parent Nodes** before you can commit the node.

- i. Click  in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the

batch sync node. For more information, see [Publish nodes](#).

8. Test the batch sync node. For more information, see [Manage auto triggered nodes](#).

2.5.5.2. MaxCompute

2.5.5.2.1. Create an ODPS SQL node

Using the SQL-like syntax, ODPS SQL nodes can process terabytes of data in distributed processing scenarios that do not require real-time processing.

Context

Generally, it takes a long time from preparing to committing a job. You can use ODPS SQL nodes to process thousands to tens of thousands of transactions. ODPS SQL nodes are online analytical processing (OLAP) applications designed to deal with large amounts of data.

Limits

- You cannot use SET statements, USE statements, or SQL alias statements independently in the code of an ODPS SQL node. They must be executed together with other SQL statements. For example, you can use a SET statement together with a CREATE TABLE statement.

```
set a=b;
create table name(id string);
```

- You cannot add comments to statements containing keywords, including SET statements, USE statements, and SQL alias statements, in the code of an ODPS SQL node. For example, the following comment is not allowed:

```
create table name(id string);
set a=b; // Comment.
create table name1(id string);
```

- The running of an ODPS SQL node during workflow development and the scheduled running of an ODPS SQL node have the following differences:
 - Running during workflow development: combines all the statements containing keywords, including SET statements, USE statements, and SQL alias statements, in the node code and executes them before executing other SQL statements.
 - Scheduled running: executes all SQL statements in sequence.

```
set a=b;
create table name1(id string);
set c=d;
create table name2(id string);
```

The following table shows the differences between the two running modes for the preceding SQL statements.

SQL statement	Running during workflow development	Scheduled running
First SQL statement	<pre>set a=b; set c=d; create table name1(id string);</pre>	<pre>set a=b; create table name1(id string);</pre>

SQL statement	Running during workflow development	Scheduled running
Second SQL statement	<pre>set a=b; set c=d; create table name2(id string);</pre>	<pre>set c=d; create table name2(id string);</pre>

- You must specify a scheduling parameter in the format of key=value. Do not add any spaces before or after the equation mark (=). Examples:

```
time = {yyyymmdd hh:mm:ss} // Incorrect format.
a =b // Incorrect format.
```

- If you use keywords such as bizdate and date as scheduling parameters, you must specify the values in the format of yyyymmdd. If you want to use other time formats, do not use the preceding keywords as scheduling parameters. Example:

```
bizdate=201908 // Incorrect format.
```

- You can only use statements starting with SELECT, READ, or WITH to query the result data for a node during the workflow development. Otherwise, no results are returned.
- Separate multiple SQL statements with semicolons (;) and place them in different lines.

- Incorrect example

```
create table1;create table2
```

- Correct example

```
create table1;
create table2;
```

Create an ODPS SQL node

- Log on to the DataWorks console.
- On the Data Analytics tab, move the pointer over  and choose **MaxCompute > ODPS SQL**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > ODPS SQL**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

- In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

- Click **Commit**.
- Edit the code of the ODPS SQL node. Edit the code of the ODPS SQL node. The code must conform to the syntax. The following example creates a table, inserts data to the table, and queries data in the table:

- i. Create a table named test1.

```
CREATE TABLE IF NOT EXISTS test1
(id BIGINT COMMENT "",
name STRING COMMENT "",
age BIGINT COMMENT "",
sex STRING COMMENT "");
```

- ii. Insert data to the table.

```
INSERT INTO test1 VALUES (1,'Zhang San',43,'Male');
INSERT INTO test1 VALUES (1,'Li Si',32,'Male');
INSERT INTO test1 VALUES (1,'Chen Xia',27,'Female');
INSERT INTO test1 VALUES (1,'Wang Wu',24,'Male');
INSERT INTO test1 VALUES (1,'Ma Jing',35,'Female');
INSERT INTO test1 VALUES (1,'Zhao Qian',22,'Female');
INSERT INTO test1 VALUES (1,'Zhou Zhuang',55,'Male');
```

- iii. Query data in the table.

```
select * from test1;
```

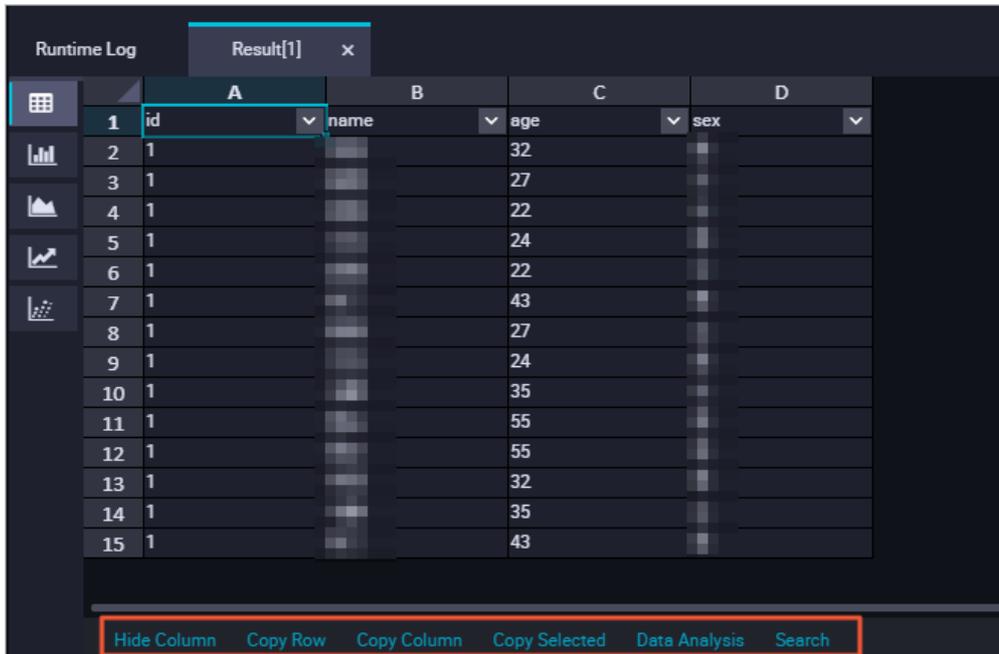
- iv. After you enter the preceding SQL statements in the code editor, click  in the toolbar. DataWorks executes your SQL statements from top to bottom and displays logs. The `INSERT INTO` statement may result in unexpected data duplication. Although DataWorks does not re-execute the `INSERT INTO` statement, it may rerun corresponding nodes. We recommend that you avoid using the `INSERT INTO` statement. When DataWorks executes the `INSERT INTO` statement, the following information appears in logs:

The `INSERT INTO` statement in SQL may cause repeated data insertion. Although SQL-level retries have been revoked for the `INSERT INTO` statement, task level retries may still happen. We recommend that you avoid the use of the `INSERT INTO` statement.

If you continue to use `INSERT INTO` statements, we deem that you are aware of the associated risks and are willing to take the consequences of potential data duplication.

- v. View the query result. DataWorks displays the query result in a workbook.

You can view or manage the query result in the workbook, or copy the query result to a local Excel file.



Action	Description
Hide Column	Select one or more columns and click Hide Column at the bottom to hide the selected columns.
Copy Row	Select one or more rows and click Copy Row at the bottom to copy the selected rows.
Copy Column	Select one or more columns and click Copy Column at the bottom to copy the selected columns.
Copy Selected	Select one or more cells in the workbook and click Copy Selected at the bottom to copy the selected cells.
Data Analysis	Click Data Analysis at the bottom to go to the workbook editing page.
Search	Click Search at the bottom to search for data in the workbook. After you click the button, a search box appears in the upper-right corner of the Results tab.

6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.

Notice You must set **Rerun** and **Parent Nodes** before you can commit the node.

- i. Click  in the toolbar.

- ii. In the **Commit Node** dialog box, ignore the alert on mismatch between the input and output that you set with those detected in code lineage analysis, enter your comments in the **Description** field, and then select **I confirm to proceed with the commission**.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the ODPS SQL node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Manage auto triggered nodes](#).

2.5.5.2.2. Create an SQL Snippet node

SQL script templates are SQL templates that involve multiple input and output parameters. Each SQL script template involves one or more source tables. You can use an SQL script template to filter, join, or aggregate data in source tables.

Context

When a new version is released for a script template, you can decide whether to upgrade the version of the script template used in your nodes to the latest version.

The script template upgrade mechanism allows developers to continuously upgrade script template versions. This mechanism enhances the process execution efficiency and optimizes the business performance.

For example, User A uses V1.0 of a script template that belongs to User B. Then, User B releases V2.0 for the script template. User A receives a notification of the new version. After User A compares the code of the two versions, User A can decide whether to upgrade the script template to the latest version.

To upgrade an SQL script template, click **Update Code** and check whether the parameter configuration of the SQL script template is valid in the new version. Set parameters for the SQL script template of the new version based on the version description. Then, save the node and commit it for deployment.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > SQL Snippet**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > SQL Snippet**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab, select a script template from the **Snippet** drop-down list. To improve development efficiency, you can create data analytics nodes by using the script templates provided by workspace members and tenants.
 - o The script templates provided by members of the current workspace are available on the **Workspace-Specific** tab.
 - o The script templates provided by tenants are available on the **Public** tab.
6. Click the **Parameters** tab in the right-side navigation pane and set parameters for the SQL script

template.

7. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
8. Commit the node.

 **Notice** You can commit the node only after you specify the Rerun and Parent Nodes parameters.

- i. Click the  icon in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

9. Test the node. For more information, see [Recurring tasks](#).

2.5.5.2.3. Create an ODPS Spark node

DataWorks supports ODPS Spark nodes. This topic uses the JAR resource type as an example to describe how to create and configure an ODPS Spark node.

Create and upload a resource

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > Resource > JAR**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Resource > JAR**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

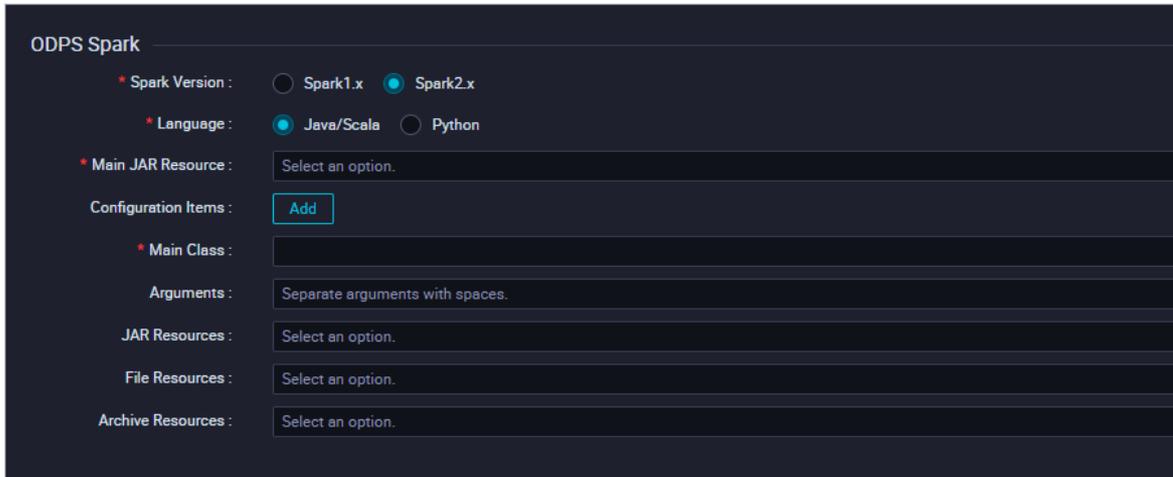
3. In the **Create Resource** dialog box, set **Resource Name** and **Location**.
4. Click **Upload** and select the target file to upload.
5. Click **OK**.

Create an ODPS Spark node

1. On the **Data Analytics** tab, move the pointer over  and choose **MaxCompute > ODPS Spark**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > ODPS Spark**.
2. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`).

3. Click **Commit**.
4. On the node configuration tab, set the parameters.



You can set **Spark Version** and **Language** as needed. The parameters vary with the value of the **Language** parameter. You can set the parameters as prompted.

The following table describes the parameters that appear after you set the Language parameter to **Java/Scala**.

Parameter	Description
Spark Version	The Spark version of the node. Valid values: Spark1.x and Spark2.x .
Language	The programming language of the node. Valid values: Java/Scala and Python . Select Java/Scala .
Main JAR Resource	The main JAR resource referenced by the node. Select a JAR resource that you uploaded from the drop-down list.
Configuration Items	The configuration items of the node. Click Add and set key and value to add a configuration item.
Main Class	The class name of the node.
Arguments	The parameter used to assign a value to a variable in the code during node scheduling. Separate multiple parameters with spaces.
JAR Resources	The JAR resource referenced by the node. Select a JAR resource that you uploaded from the drop-down list. The ODPS Spark node automatically finds the uploaded JAR resources based on the resource type.
File Resources	The file resource referenced by the node. Select a file resource that you uploaded from the drop-down list. The ODPS Spark node automatically finds the uploaded file resources based on the resource type.
Archive Resources	The archive resource referenced by the node. Select an archive resource that you uploaded from the drop-down list. The ODPS Spark node automatically finds the uploaded archive resources based on the resource type. Only compressed resources appear.

- On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
- Commit the node.

 **Notice** You can commit the node only after you specify the **Rerun** and **Parent Nodes** parameters.

- i. Click the  icon in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

7. Test the node. For more information, see [Recurring tasks](#).

2.5.5.2.4. Create a PyODPS node

DataWorks supports PyODPS nodes, which are integrated with the Python SDK of MaxCompute. You can edit Python code in PyODPS nodes of DataWorks to process data in MaxCompute.

Context

You can also use the Python SDK of MaxCompute to process data in MaxCompute.

Note

- The Python version of PyODPS nodes is 2.7.
- Each PyODPS node can process a maximum of 50 MB data and can occupy a maximum of 1 GB memory. Otherwise, DataWorks terminates the PyODPS node. Avoid writing too much data processing code for a PyODPS node.

PyODPS nodes are designed to use the Python SDK of MaxCompute. If you want to run pure Python code, you can create a Shell node to run the Python scripts uploaded to DataWorks.

Create a PyODPS node

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > PyODPS**. Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > PyODPS**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`).

4. Click **Commit**.
5. Edit the code of the PyODPS node on the node configuration tab.
 - i. Use the MaxCompute entry. Each PyODPS node includes the global variable `odps` or `o`, which is the MaxCompute entry. Therefore, you do not need to manually specify the MaxCompute entry.

```
print(odps.exist_table('PyODPS_iris'))
```

- ii. Run SQL statements. In PyODPS nodes, you can execute MaxCompute SQL statements to query data and obtain the query results. You can use the `execute_sql` or `run_sql` method to run MaxCompute job instances.

To execute statements that are not directly compatible with the MaxCompute console, you can use some methods. For example, you cannot directly execute statements other than DDL and DML in the MaxCompute console.

To execute a GRANT or REVOKE statement, use the `run_security_query` method. To run a PAI command, use the `run_xflow` or `execute_xflow` method.

```
o.execute_sql('select * from dual') # Execute the statement in synchronous mode. Other nodes are blocked until the SQL statement is executed.
instance = o.run_sql('select * from dual') # Execute the statement in asynchronous mode.
print(instance.get_logview_address()) # Obtain the Logview URL of an instance.
instance.wait_for_success() # Other nodes are blocked until the SQL statement is executed.
```

- iii. Set runtime parameters. You can use the `hints` parameter to set the runtime parameters. The type of the `hints` parameter is DICT.

```
o.execute_sql('select * from PyODPS_iris', hints={'odps.sql.mapper.split.size': 16})
```

If you set the `sql.settings` parameter for the global configuration, you must set the runtime parameters each time you run the code.

```
from odps import options
options.sql.settings = {'odps.sql.mapper.split.size': 16}
o.execute_sql('select * from PyODPS_iris') # The hints parameter is automatically set based on the global configuration.
```

- iv. Obtain SQL query results. You can use the `open_reader` method to obtain query results in the following scenarios:

- The SQL statement returns structured data.

```
with o.execute_sql('select * from dual').open_reader() as reader:
    for record in reader: # Process each record.
```

- SQL statements such as DESC are executed. In this case, you can use the `reader.raw` property to obtain raw query results.

```
with o.execute_sql('desc dual').open_reader() as reader:
    print(reader.raw)
```

 **Note** If you use a custom time variable, you must fix the variable to a time. PyODPS nodes do not support relative time variables.

6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).

7. Commit the node.

 **Notice** You can commit the node only after you specify the **Rerun** and **Parent Nodes** parameters.

- i. Click the  icon in the toolbar.

- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Recurring tasks](#).

Built-in modules for PyODPS nodes

A PyODPS node contains the following built-in modules:

- setuptools
- cython
- psutil
- pytz
- dateutil
- requests
- pyDes
- numpy
- pandas
- scipy
- scikit_learn
- greenlet
- six
- Other built-in modules in Python 2.7, such as smtplib

2.5.5.2.5. Create an ODPS Script node

You can create an ODPS Script node to develop an SQL script by using the SQL engine provided by MaxCompute V2.0.

Context

The ODPS Script node allows DataWorks to compile the SQL script as a whole, instead of compiling the SQL statements in the script one by one. In this way, the SQL script is committed and run as a whole. This guarantees that an execution plan is only queued and executed once, making full use of MaxCompute computing resources.

Create an ODPS Script node

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > ODPS Script**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > ODPS Script**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. Edit the SQL script of the ODPS Script node as required.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.

 **Notice** You can commit the node only after you specify the **Rerun** and **Parent Nodes** parameters.

- i. Click the  icon in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Recurring tasks](#).

SQL syntax and limits for ODPS Script nodes

Write SQL statements based on your business logic in a way similar to that of using a common programming language. You do not need to consider how to organize the SQL statements.

```
-- SET statements
set odps.sql.type.system.odps2=true;
[set odps.stage.reducer.num=***;]
[...]
-- DDL statements
create table table1 xxx;
[create table table2 xxx;]
[...]
-- DML statements
@var1 := SELECT [ALL | DISTINCT] select_expr, select_expr, ...
  FROM table3
  [WHERE where_condition];
@var2 := SELECT [ALL | DISTINCT] select_expr, select_expr, ...
  FROM table4
  [WHERE where_condition];
@var3 := SELECT [ALL | DISTINCT] var1.select_expr, var2.select_expr, ...
  FROM @var1 join @var2 on ...;
INSERT OVERWRITE|INTO TABLE [PARTITION (partcol1=val1, partcol2=val2 ...)]
  SELECT [ALL | DISTINCT] select_expr, select_expr, ...
  FROM @var3;
[@var4 := SELECT [ALL | DISTINCT] var1.select_expr, var.select_expr, ... FROM @var1
  UNION ALL | UNION
  SELECT [ALL | DISTINCT] var1.select_expr, var.select_expr, ... FROM @var2;
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] table_name
  AS
  SELECT [ALL | DISTINCT] select_expr, select_expr, ...
  FROM var4;]
```

SQL syntax and limits for ODPS Script nodes

- ODPS Script nodes support SET statements, DML statements, and some DDL statements. The DDL statements used to return data, such as DESC and SHOW statements, are not supported.
- A complete script consists of SET statements, DDL statements, and DML statements in sequence. You can write one or more statements of each type, or even skip a type without writing any statements of that type. However, you cannot mix different types of statements together. You must strictly follow the sequence of SET statements > DDL statements > DML statements.
- The at signs (@) residing before some statements indicate that these statements are connected by using variables.
- A script supports only one statement that returns data, such as an independent SELECT statement. If multiple such statements are provided, an error occurs. We recommend that you do not use SELECT statements in a script.
- A script supports only one CREATE TABLE AS statement, which must be the last statement. We recommend that you put CREATE TABLE statements and INSERT statements in different sections to separate them.
- If one statement in a script fails, the whole script fails.
- A job is generated to process data only after all the input data is prepared for a script.
- If a script writes data to a table and then reads the table, an error occurs. For example, an error occurs for the following statements:

```
insert overwrite table src2 select * from src where key > 0;
@a := select * from src2;
select * from @a;
```

To avoid the error, modify the statements to the following:

```
@a := select * from src where key > 0;
insert overwrite table src2 select * from @a;
select * from @a;
```

Sample script:

```
create table if not exists dest(key string , value bigint) partitioned by (d string);
create table if not exists dest2(key string,value bigint ) partitioned by (d string);
@a := select * from src where value >0;
@b := select * from src2 where key is not null;
@c := select * from src3 where value is not null;
@d := select a.key,b.value from @a left outer join @b on a.key=b.key and b.value>0;
@e := select a.key,c.value from @a inner join @c on a.key=c.key;
@f := select * from @d union select * from @e union select * from @a;
insert overwrite table dest partition (d='20171111') select * from @f;
@g := select e.key,c.value from @e join @c on e.key=c.key;
insert overwrite table dest2 partition (d='20171111') SELECT * from @g;
```

Scenarios of ODPS Script nodes

- You can use an ODPS Script node to rewrite a single statement with nested subqueries, or a script that must be split into multiple statements due to its complexity.
- Data from different data stores may be prepared at different time points, and the time difference may be large. For example, the data from one data store can be prepared at 01:00, whereas that from another data store can be prepared at 07:00. In this case, table variables are not suitable for connecting statements. You can use an ODPS Script node to combine the statements to a script.

2.5.5.2.6. Create an ODPS MR node

MaxCompute supports the MapReduce API. You can create and commit ODPS MR nodes that call the Java API operations of MapReduce to develop MapReduce programs for processing data in MaxCompute.

Context

Before you create ODPS MR nodes, you must upload, commit, and then deploy required resources.

Create a resource

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > Resource > JAR**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Resource > JAR**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Resource** dialog box, set **Resource Name** and **Location**.

 **Note**

- The resource name can be different from the name of the uploaded file.
- Convention for naming resources: A resource name can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive and must be 1 to 128 characters in length. A JAR resource name must end with `.jar`. A Python resource name must end with `.py`.

4. Click **Upload** and select the target file to upload.
5. Click **OK**.
6. Click  in the toolbar to commit the resource to the development environment.

Create an ODPS MR node

1. On the Data Analytics tab, find the target workflow, right-click **MaxCompute**, and then choose **Create > ODPS MR**.
2. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`).

3. Click **Commit**.
4. Edit the ODPS MR node.

```

-- Create an input table.
CREATE TABLE if not exists jingyan_wc_in (key STRING, value STRING);
-- Create an output table.
CREATE TABLE if not exists jingyan_wc_out (key STRING, cnt BIGINT);
--- Create the dual table.
drop table if exists dual;
create table dual(id bigint); -- Create the dual table if no dual table exists in the current workspace and initialize the table.
--- Initialize the dual table.
insert overwrite table dual select count(*) from dual;
--- Insert the sample data to the wc_in table.
insert overwrite table jingyan_wc_in select * from (
select 'project','val_pro' from dual
union all
select 'problem','val_pro' from dual
union all
select 'package','val_a' from dual
union all
select 'pad','val_a' from dual
) b;
-- Reference the uploaded JAR package. You can find the JAR package in the resource list, right-click the JAR resource, and select Insert Resource Path.
--@resource_reference{"mapreduce-examples.jar"}
jar -resources mapreduce-examples.jar -classpath ./mapreduce-examples.jar com.aliyun.odps.mapred.open.example.WordCount jingyan_wc_in jingyan_wc_out

```

Pay attention to the following information when you write the code:

- `--@resource_reference` : references a resource. Find the target resource, right-click it, and then select **Insert Resource Path** to generate the reference statement.
 - `-resources` : the name of the referenced JAR resource.
 - `-classpath` : the path of the JAR resource. You can enter `./Resource name` because the resource has been referenced.
 - `com.aliyun.odps.mapred.open.example.WordCount` : the main class in the JAR resource to be called during node running. It must be the same as the main class name in the JAR resource.
 - `jingyan_wc_in` : the name of the input table of the ODPS MR node. The input table is created in the preceding code.
 - `jingyan_wc_out` : the name of the output table of the ODPS MR node. The output table is created in the preceding code.
 - If you use multiple JAR resources in a single ODPS MR node, separate the resource paths with commas (,), for example, `-classpath ./xxx1.jar,./xxx2.jar`.
5. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
 6. Commit the node.

 **Notice** You can commit the node only after you specify the **Rerun** and **Parent Nodes** parameters.

- i. Click the  icon in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.

iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

7. Test the node. For more information, see [Recurring tasks](#).

2.5.5.2.7. Create a MaxCompute table

This topic describes how to create a MaxCompute table.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > Table**. Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Table**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Table** dialog box, set **Table Name** and click **Commit**.

 **Notice** A table name can be up to 64 characters in length. The table name must start with a letter and cannot contain Chinese or special characters.

4. On the table configuration tab that appears, set the parameters in the **General** section.

Parameter or button	Description
Display Name	The display name of the table.
Level 1 Folder	The name of the level-1 folder where the table resides.  Note Level-1 and level-2 folders only show the table locations in DataWorks so that you can better manage tables.
Level 2 Folder	The name of the level-2 folder where the table resides.
Create Folder	Goes to the Folder Management tab. On this tab, you can create level-1 and level-2 folders for tables.
Description	The description of the table.

5. Create a table. Use one of the following methods to create a table:

- o Create a table by using DDL statements.

Click **DDL Statement** in the top navigation bar. In the dialog box that appears, enter the statements for creating a table.

After you finish editing the statements, click **Generate Table Schema**. Information is automatically entered in the General, Physical Model, and Schema sections.

- Create a table on the graphical user interface (GUI).

If DDL statements are inappropriate for you to create a table, try to use the GUI. The following table describes the relevant parameters for creating a table on the GUI.

Section	Parameter or button	Description
Physical Model	Partitioning	Specifies whether the table is partitioned. Valid values: Partitioned Table and Non-Partitioned Table .
	Time-to-Live	The time-to-live of data in MaxCompute. If you select this check box, you must enter a number in the TTL field. If the table or partition is stored for more than the specified number of days, data that has not been updated is cleared.
	Table Level	The level of the table. Generally, tables are divided into operation data store (ODS), common data model (CDM), and application data service (ADS) levels. You can specify a custom level name.
	Categories	<p>The category of the table. Tables are categorized into basic services, advanced services, and other services. You can specify a custom category name.</p> <p>If you want to create a table category or level, click Create Level to go to the Level Management tab.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note Categories are designed only for your management convenience and do not involve underlying implementation.</p> </div>
	Table Type	The type of the table. Default value: Internal Table .
	Field Name	The name of the field. The name can contain letters, digits, and underscores (_).
	Display Name	The display name of the field.
	Data Type	The data type of the field.
	Definition or Maximum Value Length	The maximum value length of a field. You can set a maximum value length only for fields of the DECIMAL, VARCHAR, ARRAY, MAP, and STRUCT types.
	Description	The description of the field.
	Primary Key Field	Specifies whether the field serves as the primary key or part of a composite primary key.
	Create Field	Adds a field to the table.

Section	Parameter or button	Description
Schema	Delete Field	Deletes a field from the table.  Note If you delete a field from an existing table, DataWorks requests you to delete the table and create another table with the same name. This operation is forbidden in the production environment.
	Move Up	Adjusts the field sequence of the table. If you adjust the sequence of fields in an existing table, DataWorks requests you to delete the table and create another table with the same name. This operation is forbidden in the production environment.
	Move Down	The description is the same as that of the Move Up operation.
	Add	Adds a partition to the table. If you add a partition to an existing table, DataWorks requests you to delete the table and create another table with the same name. This operation is forbidden in the production environment.
	Delete	Deletes a partition from the table. If you delete a partition from an existing table, DataWorks requests you to delete the table and create another table with the same name. This operation is forbidden in the production environment.
	Actions	Commits a partition or deletes a field.
Partition Field Design	Add	Adds a partition field.
	Data Type	The data type of the partition field. We recommend that you use the STRING type for all partition fields.
	Length	The maximum length of the partition field. You can set the maximum length only for fields of the VARCHAR-type.
	Description	The description of the partition field.
	Partition Column Date Format	The format of the date partition. If the partition field is a date, although the data type may be STRING, select or enter a date format, such as <i>yyyymmdd</i> or <i>yyyy-mm-dd</i> .
	Partition Column Date Granularity	The granularity of the date partition. The granularities can be second, minute, hour, day, month, quarter, and year. You can enter a partition granularity as required. If you want to specify multiple partition granularities, note that a greater granularity corresponds to a higher partition level. For example, three partitions whose granularities are day, hour, and month, respectively, are available. Multi-level partitions are in the hierarchical order of level-1 partition (month), level-2 partition (day), and level-3 partition (hour).

 **Note** This section is available only when Partitioning under Physical Model is set to Partitioned Table.

- Click **Commit in Development Environment** and **Commit to Production Environment** in sequence. If you are using a workspace in basic mode, you only need to click **Commit to Production Environment**.

Button	Description
Load from Development Environment	<p>If the table has been committed to the development environment, the button is clickable. After you click the button, the information about the table you create in the development environment overwrites the table information on the current page.</p> <p> Note This feature is supported only for MaxCompute tables.</p>
Commit in Development Environment	<p>Before you click the button, make sure that you have filled in all required parameters on the table configuration tab. Do not click the button if any parameters are not specified.</p>
Load from Production Environment	<p>After you click the button, the information about the table that is committed to the production environment overwrites the table information on the current page.</p> <p> Note This feature is supported only for MaxCompute tables.</p>
Commit to Production Environment	<p>After you click the button, the table is created in the workspace of the production environment.</p>

What's next

After the table is created, you can query the table data and modify or delete the table. For more information, see [Manage tables](#).

2.5.5.2.8. Create, reference, and download resources

This topic describes how to create, reference, and download JAR and Python resources.

Context

If your code or function requires resource files such as .jar files, you can upload resources to your workspace and reference them.

If the existing built-in functions do not meet your requirements, DataWorks allows you to create user-defined functions (UDFs) and customize processing logic. You can upload the required JAR packages to your workspace so that you can reference them when you create UDFs.

Note

- You can view built-in functions on the **Built-In Functions** tab. For more information, see [View built-in functions](#).
- You can view the UDFs that you have committed or deployed on the **MaxCompute Functions** tab.

The resources that you can upload to MaxCompute include text files, MaxCompute tables, Python code, and compressed packages in the .zip, .tgz, .tar.gz, .tar, and .jar formats. You can read or use these resources when you run UDFs or MapReduce.

MaxCompute provides API operations for you to read and use resources. The following types of MaxCompute resources are available:

- Python:** the Python code you have written. You can use Python code to register Python UDFs.

- **JAR**: the compiled Java JAR packages.
- **Archive**: the compressed files that can be identified by the file name extension. Supported file types include .zip, .tgz, .tar.gz, .tar, and .jar.
- **File**: files in the .zip, .so, or .jar format.

JAR resources and file resources have the following differences:

- To create a JAR resource, write Java code in the offline Java environment, compress the code to a JAR package, and upload the package as a JAR resource to DataWorks.
- To create a file resource that is smaller than or equal to 500 KB in size, you can create and edit it in the DataWorks console.
- To create a file resource that is larger than 500 KB in size, select **Large File (more than 500 KB)** and click Upload to upload the file.

 **Note** Each resource file to be uploaded in the DataWorks console cannot exceed 30 MB.

Create a JAR resource

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **MaxCompute > Resource > JAR**.
Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Resource > Python**.

 **Notice** The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

3. In the **Create Resource** dialog box, set **Resource Name** and **Location**.

 **Note**

- The resource name can be different from the name of the uploaded file.
- A resource name can contain letters, digits, underscores (_), and periods (.), and is not case-sensitive. It must be 1 to 128 characters in length. A JAR resource name must end with .jar, and a Python resource name must end with .py.

4. Click **Upload** and select the target file to upload.
5. Click **OK**.
6. Click  in the toolbar to commit the resource to the development environment.

Create a Python resource and register a UDF

1. Create a Python resource.
 - i. On the **Data Analytics** tab, move the pointer over  and choose **MaxCompute > Resource > Python**. Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Resource > Python**.
 - ii. In the **Create Resource** dialog box, set **Resource Name** and **Location**.
 - iii. Click **OK**.

- iv. On the configuration tab that appears, edit the code of the created resource. Sample code:

```
from odps.udf import annotate
@annotate("string->bigint")
class ipint(object):
    def evaluate(self, ip):
        try:
            return reduce(lambda x, y: (x << 8) + y, map(int, ip.split('.')))
        except:
            return 0
```

- v. Click  in the toolbar.

2. Register a UDF.

- i. On the **Data Analytics** tab, move the pointer over  and choose **MaxCompute > Function**. Alternatively, you can click a workflow in the Business Flow section, right-click **MaxCompute**, and then choose **Create > Function**.
- ii. In the **Create Function** dialog box, set **Function Name** and **Location**.
- iii. Click **Commit**.
- iv. In the **Register Function** section of the configuration tab that appears, enter the class name and the name of the Python resource that has been created, and then click  in the toolbar. In this example, the class name is `ipint.ipint`.
- v. Check whether the ipint function is valid and meets your expectation. For example, you can create an ODPS SQL node to test the ipint function by running an SQL statement.

Reference and download resources

- For more information about how to reference resources for functions, see [Register a UDF](#).
- For more information about how to reference resources for nodes, see [Create an ODPS MR node](#).

To download a resource, double-click **Resource** under the target workflow. In the resource list that appears, move the pointer over the required resource and click **Download**.

2.5.5.2.9. Register a UDF

DataWorks allows you to develop UDFs in Python and Java. This topic describes how to register a UDF.

Prerequisites

Before you register a UDF, you must upload the related resource.

Procedure

1. Log on to the DataWorks console.
2. Create a workflow. For more information, see [Create a workflow](#).
3. Write Java code in the offline Java environment, compress the code to a JAR package, and upload the package as a JAR resource to DataWorks. For more information, see [Create a JAR resource](#).
4. Create a UDF.
 - i. Find the target workflow, right-click **MaxCompute**, and then choose **Create > Function**.
 - ii. In the **Create Function** dialog box, set **Function Name** and **Location** and click **OK**.

iii. In the **Register Function** section of the configuration tab that appears, set the parameters.

Parameter	Description
Function Type	The type of the function. Valid values: Mathematical Function, Aggregate Function, String Function, Date Function, Analytic Function, and Other.
Engine Instance MaxCompute	The MaxCompute engine instance bound to the current workspace. By default, you cannot change the engine instance.
Function Name	The name of the function, which is used to reference the function in SQL. The function name must be globally unique and cannot be modified after the function is registered.
Owner	The owner of the function. By default, this parameter is automatically set.
Class Name	Required. The name of the class for implementing the function. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note If the resource type is Python, enter the class name in the Python resource name. Class name format. Do not include the .py extension in the resource name.</p> </div>
Resources	Required. The list of resources. You can search for existing resources in the current workspace in fuzzy match mode.
Description	The description of the function.
Expression Syntax	The instructions on how to use the function, for example, <code>test</code> .
Parameter Description	The description of supported input and output parameter types.
Return Value	Optional. The value to return. Example: 1.
Example	Optional. An example of the function.

5. Click  in the toolbar.

6. Commit the function.

i. Click  in the toolbar.

ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.

iii. Click **OK**.

2.5.5.3. EMR

2.5.5.3.1. Create an EMR MR node

You can create an EMR MR node to compute a large-scale dataset by using multiple Map tasks in a parallel manner.

Prerequisites

The EMR folder is available on the DataStudio page only after you bind an E-MapReduce compute engine to the current workspace on the **Project Management** page.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **EMR > EMR MR**. Alternatively, you can click a workflow in the Business Flow section, right-click **EMR**, and then choose **Create > EMR MR**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab, select an E-MapReduce compute engine from the **Engine Instance EMR** drop-down list and edit the code of the node.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.3.2. Create an EMR Spark SQL node

You can create an EMR Spark SQL node to use the distributed SQL query engine to process structured data, improving the task execution efficiency.

Prerequisites

The EMR folder is available on the DataStudio page only after you bind an E-MapReduce compute engine to the current workspace on the **Project Management** page.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **EMR > EMR Spark SQL**. Alternatively, you can click a workflow in the Business Flow section, right-click **EMR**, and then choose **Create > EMR Spark SQL**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.

5. On the node configuration tab, select an E-MapReduce compute engine from the **Engine Instance EMR** drop-down list and edit the code of the node.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).
8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.3.3. Create an EMR Spark node

You can create an EMR Spark node to perform complex memory analysis and build large and low-latency data analysis applications.

Prerequisites

The EMR folder is available on the DataStudio page only after you bind an E-MapReduce compute engine to the current workspace on the **Project Management** page.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **EMR > EMR Spark**. Alternatively, you can click a workflow in the Business Flow section, right-click **EMR**, and then choose **Create > EMR Spark**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab, select an E-MapReduce compute engine from the **Engine Instance EMR** drop-down list and edit the code of the node.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).
8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.3.4. Create an EMR Hive node

This topic describes how to create an EMR Hive node. This type of node allows you to use SQL-like statements to read data from, write data to, and manage data warehouses with a large amount of data stored in a distributed storage system. By using this type of node, you can efficiently analyze a large amount of log data.

Prerequisites

The EMR folder is available on the DataStudio page only after you bind an E-MapReduce compute engine to the current workspace on the **Project Management** page.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **EMR > EMR Hive**. Alternatively, you can click a workflow in the Business Flow section, right-click **EMR**, and then choose **Create > EMR Hive**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab, select an E-MapReduce compute engine from the **Engine Instance EMR** drop-down list and edit the code of the node.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).
8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.4. Hologres

2.5.5.4.1. Create a Hologres SQL node

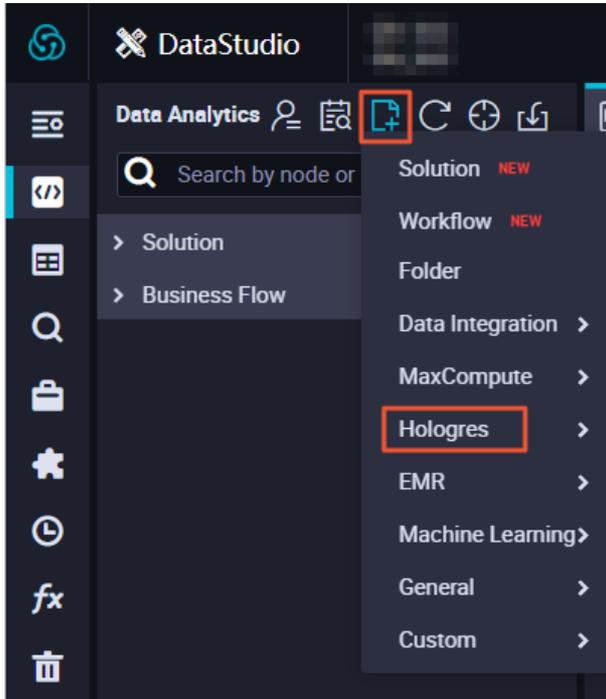
This topic describes how to create a Hologres SQL node. Hologres seamlessly integrates with MaxCompute at the underlying layer. This integration allows you to use standard PostgreSQL statements to query and analyze large volumes of data stored in MaxCompute. In this case, you do not need to transfer data. This allows you to quickly obtain query results.

Prerequisites

A **Hologres** compute engine instance is added on the **Project Management** page. This ensures that the **Hologres** folder is displayed on the page on which you want to create a Hologres SQL node.

Procedure

1. Log on to the DataWorks console.
2. Move the pointer over the  icon and choose **Hologres > Hologres SQL**.



3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

 **Note** The name of the node must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the tab that appears, edit and run the code in the code editor. After the Hologres SQL node is created, edit the code in compliance with the required SQL syntax.
6. On the configuration tab of the node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure properties for the node. For more information, see [Basic properties](#).
7. Save and commit the node.

 **Notice** You must set the **Rerun** and **Parent Nodes** parameters before you commit the node.

- i. Click the  icon in the top toolbar to save the node.
- ii. Click the  icon in the top toolbar.
- iii. (Optional) In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iv. Click **OK**.

If the workspace that you use is in standard mode, you must click **Deploy** in the top navigation bar after you commit the node. For more information, see [Deploy nodes](#).

8. Test the node. For more information, see [Manage auto triggered nodes](#).

2.5.5.5. AnalyticDB for PostgreSQL

2.5.5.5.1. Create an AnalyticDB for PostgreSQL node

You can create AnalyticDB for PostgreSQL nodes in the DataWorks console to build an online extract, transform, load (ETL) process.

Prerequisites

The AnalyticDB for PostgreSQL compute engine is associated with the workspace in which you want to create an AnalyticDB for PostgreSQL node. The AnalyticDB for PostgreSQL folder is displayed in a workspace only after you associate an AnalyticDB for PostgreSQL compute engine instance with the workspace on the **Workspace Management** page. For more information, see [Create an AnalyticDB for PostgreSQL node](#).

Procedure

1. [Log on to the DataWorks console](#).
2. On the **DataStudio** page, move the pointer over the  icon and choose **AnalyticDB > ADB for PostgreSQL**. Alternatively, you can find the required workflow, right-click the workflow name, and then choose **Create > AnalyticDB for PostgreSQL > ADB for PostgreSQL**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The name of the node must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. Configure the AnalyticDB for PostgreSQL node.
 - i. Select a data source from the **Select a connection** drop-down list.

Notice

- When you associate the AnalyticDB for PostgreSQL compute engine with the workspace, DataWorks automatically creates an AnalyticDB for PostgreSQL data source.
- You can select a data source that is added only by using the connection string mode.

- ii. Compile SQL statements. After you select a data source, compile SQL statements based on the syntax that is supported by AnalyticDB for PostgreSQL.
 - iii. Click the  icon in the top toolbar to save the SQL statements.
 - iv. Click the  icon in the top toolbar to execute the SQL statements.
6. Click **Properties** in the right-side navigation pane. In the Properties panel, configure scheduling properties for the node. For more information, see [Basic properties](#).
 7. Save and commit the node.

 **Notice** You must specify the **Rerun** and **Parent Nodes** parameters in the Properties panel before you commit the node.

- i. Click  in the top toolbar to save the node.
- ii. Click the  icon in the top toolbar.
- iii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.

iv. Click **OK**.

If the workspace that you use is in standard mode, you must click **Deploy** in the upper-right corner after you commit the AnalyticDB for PostgreSQL node. For more information, see [Deploy nodes](#).

8. Test the node. For more information, see [Manage auto triggered nodes](#).

2.5.5.5.2. Create an AnalyticDB for PostgreSQL table

This topic describes how to create an AnalyticDB for PostgreSQL table.

Prerequisites

- An AnalyticDB for PostgreSQL instance is associated with the workspace in which you want to create an AnalyticDB for PostgreSQL table. The AnalyticDB for PostgreSQL folder is displayed on the DataStudio page of a workspace only after you associate an AnalyticDB for PostgreSQL instance with the workspace on the **Project Management** page. For more information, see [Create an AnalyticDB for PostgreSQL table](#).
- The metadata of the associated AnalyticDB for PostgreSQL instance is collected on the **Data Map** page. For more information, see [Collect metadata from an AnalyticDB for PostgreSQL data source](#).

Procedure

1. [Log on to the DataWorks console](#).
2. On the **DataStudio** page, move the pointer over the  icon and choose **AnalyticDB > table**.
Alternatively, you can find your workflow in the Business Flow section, right-click **AnalyticDB for PostgreSQL**, and then choose **Create > ADB visual table creation**.
3. In the **Create Table** dialog box, set **Table Name**.

Notice

- The table name must be in the format of `schema_name.table_name`.
- The values of `schema_name` and `table_name` must be 1 to 63 characters in length and can contain letters, digits, and underscores (`_`). The values must start with a letter or underscore (`_`).
- If you associate multiple AnalyticDB for PostgreSQL instances with the current workspace, you must select one based on your business requirements.

4. Click **Commit**. The table configuration tab appears. The upper part of the table configuration tab displays the table name and AnalyticDB for PostgreSQL instance name.
5. In the **General** section, set the parameters.

Parameter	Description
Level 1 theme	The name of the level-1 folder where the table resides.  Note Level-1 and level-2 folders show the table locations in DataWorks for you to manage tables more conveniently.
Level 2 theme	The name of the level-2 folder where the table resides.

Parameter	Description
Create a theme	<p>Click Create a theme to go to the Folder Management tab. On this tab, you can create level-1 and level-2 folders for tables.</p> <p>After you create a folder, click the  icon next to Create Folder to synchronize the folder.</p>
Description	The description of the table.

6. In the **Physical model design** section, set the parameters.

Parameter	Description
Level selection	The layer where the table data is stored or processed. A data warehouse consists of the operational data store (ODS), common data model (CDM), and application data store (ADS) layers. You can customize a name for each layer.
Physical classification	<p>The category of the table. Tables are categorized into basic services, advanced services, and other services. You can customize a name for each category.</p> <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note Categories are designed only for your management convenience and do not involve underlying implementation.</p> </div>
New Level	The levels and categories that you want to create. To add levels and categories, click New Level to go to the Hierarchical management tab. After levels and categories are created, click the  icon.

7. In the **AnalyticDB for PostgreSQL table design** section, set the parameters. You can configure the schema of an AnalyticDB for PostgreSQL table on the following tabs: **Column information settings**, **Index settings**, **Sub-table design**, and **Partition settings (optional)**.

Tab	Parameter	Description
Column information settings	New columns	Allows you to click the button and set the relevant parameters to create a field.
	Name	The name of the field.
	Field type	The data type of the field.
	Field length	The length of the field. You can specify the length for fields only of some specific data types.
	Default value	The default value of the field.
	Allow to be empty	Specifies whether the field can be empty.
	Is it the primary key?	Specifies whether the field serves as the primary key.
	Foreign key	Specifies whether the field serves as a foreign key.

Tab	Parameter	Description
	Operation	<ul style="list-style-type: none"> You can perform the following operations on a new field: save, cancel, delete, move up, and move down. You can perform the following operations on an existing field: modify, delete, move up, and move down.
Index settings	New columns	Allows you to click the button and set the relevant parameters to create an index.
	Index name	The name of the index. Make sure that you specify a unique name.
	Include columns	<p>The field on which the index will be created. To select a field, click Edit. In the Select at least one index dialog box, click the + icon. All the created fields appear in the Column information drop-down list.</p> <p>Select the field from the Column information drop-down list and click Save.</p>
	Index type	The type of the index. Valid values: Normal , Primary Key , and Unique .
	Index mode	The mode for indexing data in the fields. Valid values: B-tree , Bitmap , and GiST .
	Operation	<ul style="list-style-type: none"> You can perform the following operations on a new index: save, cancel, delete, move up, and move down. You can perform the following operations on an existing index: modify, delete, move up, and move down.
Sub-table design	Hash (Recommended) , Copy Schema , and Random (Not Recommended)	<p>The way in which the partition key is generated. Take Hash (Recommended) as an example. Click New columns and select the target field from the Name drop-down list. The information about the selected field appears. Click Save.</p> <p>For more information, see the Column information settings section of this table.</p>
Partition settings (optional)	Partition settings (optional)	The partitions of the table. You can configure the partitions based on your business requirements.

- Click **Submit to development environment** and **Submit to production environment** in sequence. If you are using a workspace in basic mode, you need only to click **Submit to production environment**.
- In the **Submit changes** dialog box, confirm that the table creation statements are correct, select a resource group from the **Select a resource group** drop-down list, and then click **Confirm execution**.

What's next

After the AnalyticDB for PostgreSQL table is created, you can query the table data, modify the table, or delete the table. For more information, see [Manage tables](#).

2.5.5.6. Algorithm

2.5.5.6.1. Create a PAI node

PAI nodes are used to call tasks that are created on PAI and schedule production activities based on the node configuration.

Prerequisites

To create a PAI node in DataWorks, you must first create a PAI experiment in PAI.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **Machine Learning > PAI Experiment**.
Alternatively, you can click a workflow in the Business Flow section, right-click **Algorithm**, and then choose **Create > PAI Experiment**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. Select the PAI experiment that you have created from the **Experiment** drop-down list and load it. If you want to modify the PAI experiment, click **Edit in PAI Console**.
6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).
8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.7. General

2.5.5.7.1. Create a for-each node

This topic describes how to use a for-each node to repeat a loop twice and display the loop count.

Prerequisites

The MaxCompute module is available on the DataStudio page only after you bind a MaxCompute compute engine to the current workspace on the **Project Management** page.

Context

You can use a for-each node to repeat a loop for a maximum of 128 times. If the loop count exceeds this limit, an error occurs.

If the for-each node needs to perform logic judgment and result traversal, you can use the branch node. However, the branch node must be used with the merge node for result traversal.

Procedure

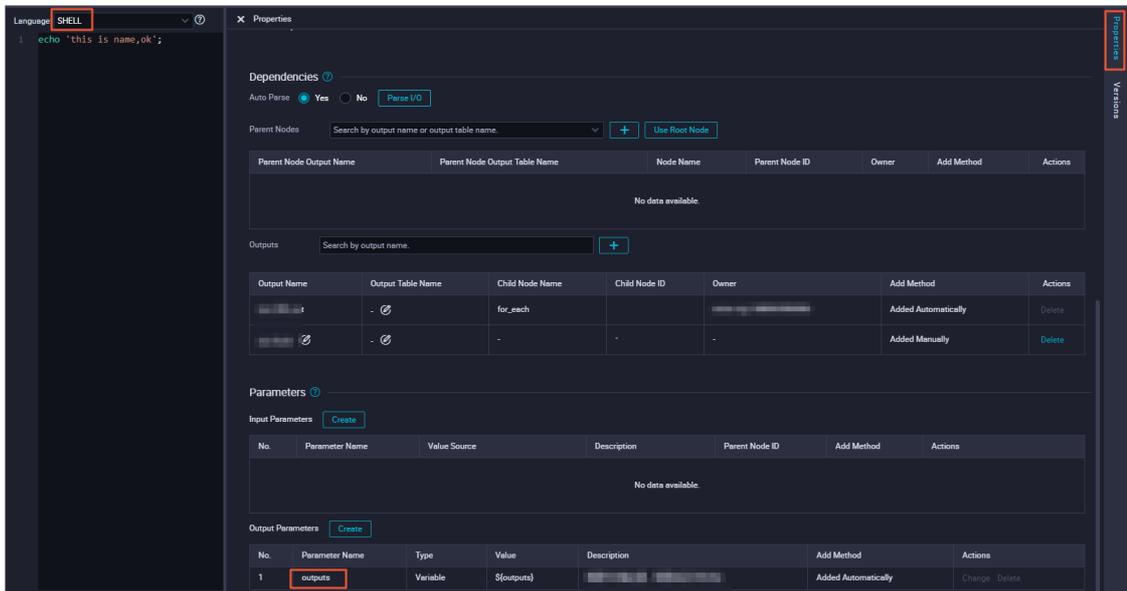
1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **General > for-each**. Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > for-each**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. Create a workflow with an assignment node as the parent node and a for-each node as the child node.
For more information, see [Create a workflow](#).
 - i. Double-click the created assignment node. Set the language of the assignment node to SHELL, and enter the following code:

```
echo 'this is name,ok';
```

On the node configuration tab, click the **Properties** tab in the right-side navigation pane. By default, the **outputs** parameter appears in the **Output Parameters** section.



- ii. Double-click the created for-each node. Enter the following code for the for-each node:

```
echo ${dag.loopTimes} ----Display the loop count.
```

Note

- The start and end nodes of the for-each node have fixed logic and cannot be edited.
- After you modify the code of the Shell node, save the modification. No message will appear to remind you to save the modification when you commit the node. If you do not save the modification, the code cannot be updated to the latest version in time.

A for-each node supports the following environment variables:

- `${dag.foreach.current}`: the current data row.
- `${dag.loopDataArray}`: the input dataset.
- `${dag.offset}`: the offset of the loop count to 1.
- `${dag.loopTimes}`: the loop count, whose value equals to the value of `${dag.offset}` plus 1.

```
// Compare the code of the Shell node with that of a common for loop.
data=[] // It is equivalent to ${dag.loopDataArray}.
// i is equivalent to ${dag.offset}.
for(int i=0;i<data.length;i++) {
    print(data[i]); // data[i] is equivalent to ${dag.foreach.current}.
}
```

The `${dag.loopDataArray}` parameter is the default input parameter of the for-each node. Set this parameter to the value of the outputs parameter of the parent node. If you do not set this parameter, an error occurs when you commit the node.

6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.

 **Notice** You can commit the node only after you specify the **Rerun** and **Parent Nodes** parameters.

- i. Click the  icon in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Publish** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Recurring tasks](#).

2.5.5.7.2. Create a do-while node

You can define mutually dependent nodes, including a loop decision node named end, in a do-while node. DataWorks repeatedly runs the nodes and exits the loop only when the end node returns False.

Context

Note A loop can be repeated for a maximum of 128 times. If the loop count exceeds this limit, an error occurs.

The do-while node supports the MaxCompute SQL, SHELL, and Python languages. If you use MaxCompute SQL, you can use a **CASE WHEN** statement to evaluate whether the specified condition for exiting the loop is met.

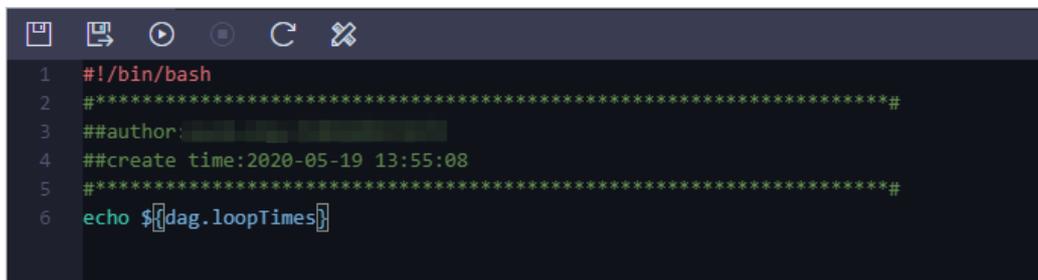
Simple example

This section describes how to use a do-while node to repeat a loop five times and display the loop count each time the loop runs.

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **General > do-while**. Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > do-while**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

Note The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (**_**), and periods (**.**).

4. Click **Commit**.
5. Define the loop body. By default, the do-while node consists of the start, SQL, and end nodes.
 - o The start node marks the start up of a loop and does not have any business effect.
 - o DataWorks provides the SQL node as a sample business processing node. You must replace the SQL node with your own business processing node, for example, a Shell node named Display loop count.



```
1 #!/bin/bash
2 #*****#
3 ##author: *****#
4 ##create time:2020-05-19 13:55:08
5 #*****#
6 echo ${dag.loopTimes}
```

- o The end node marks the end of a loop and determines whether to start the loop again. In this example, it defines the condition for exiting the loop for the do-while node.

The end node is an assignment node. It generates only True or False, indicating whether to start the loop again or exit the loop.

The `${dag.loopTimes}` variable is used in both the Display loop count node and the end node. It is a reserved variable of DataWorks. This variable indicates the loop count and the value increments from 1. All internal nodes of the do-while node can reference this variable.

In the code shown in the preceding figure, the value of the `dag.loopTimes` variable is compared with 5 to limit the loop count. The value of the `dag.loopTimes` variable is 1 when the loop runs for the first time and is incremented by 1 each time, for example, 2 for the second time. In the fifth loop, the value is 5. In this case, the result of `${dag.loopTimes}<5` is False, and the do-while node exits the loop.

6. Run the do-while node. You can configure the scheduling properties for the do-while node as needed and commit it to **Operation Center** for running.
 - o do-while node: The do-while node appears as a whole node in Operation Center. To view the loop details about the do-while node, right-click the node in the DAG and select **View Internal Nodes**.

- Internal loop body: This view is divided into three parts.
 - The left pane of the view lists the rerun history of the do-while node. A record is generated each time a do-while node instance is run.
 - The middle pane of the view shows a loop record list. A record is generated each time the loop of the do-while node is run. The running status of each loop also appears.
 - The right pane of the view shows the details about the do-while node each time the loop is run. You can click a record in the loop record list to view the running details.
7. View the running result. View the internal loop body. In the loop record list, click the record corresponding to the third loop. The loop count is 3 in the runtime logs.

You can also view the runtime logs of the end node that are generated when the loop runs for the third time and for the fifth time, respectively.

Based on the preceding simple example, the do-while node works in the following way:

- i. Run from the start node.
- ii. Run nodes in sequence based on the defined node dependencies.
- iii. Define the condition for exiting the loop in the end node.
- iv. Run the conditional statement of the end node after the loop ends for the first time.
- v. Record the loop count as 1 and start the loop again if the conditional statement returns True in the runtime logs of the end node.
- vi. Exit the loop if the conditional statement returns False in the runtime logs of the end node.

Complex example

In addition to simple scenarios, do-while nodes can also be used in complex scenarios where each row of data is processed in sequence by using a loop. Before you process data in such scenarios, make sure that:

- You have deployed a parent node that can export queried data to the do-while node. You can use an assignment node to meet this condition.
- The do-while node can obtain the output of the parent node. You can configure the node context and dependencies to meet this condition.
- The internal nodes of the do-while node can reference each row of data. In this example, the existing node context is enhanced and the system variable `#{dag.offset}` is used to reference the context of the do-while node.

This section describes how to use the do-while node to display the data entries in a table in sequence until all data entries in the table are displayed. Each time the loop runs, a data entry is displayed.

1. On the **Data Analytics** tab, double-click the created do-while node.
2. Define the loop body.
 - i. Create an assignment node named Initialize dataset and add it as the parent node of the do-while node. The parent node generates a test dataset.
 - ii. On the Properties tab of the do-while node, define an input parameter in the **Parameters** section. Set Parameter Name to input and Value Source to the output of the parent node.

- iii. Write code for the business processing node named Print each data row.
 - `${dag.offset}` : a reserved variable of DataWorks. This variable indicates the offset of the loop count to 1. For example, the offset is 0 when the loop runs for the first time and 1 for the second time. The offset equals to the loop count minus 1.

- `${dag.input}` : the context that you configure for the do-while node. In the preceding steps, the input parameter is defined for the do-while node and the value of the input parameter is the output of the parent node named Initialize dataset.

The internal nodes of the do-while node can directly use `${dag.${ctxKey}}` to reference the context. In this example, `ctxKey` is set to input. Therefore, you can use `${dag.input}` to reference the context.

- `${dag.input[${dag.offset}]}` : the data obtained from the table generated by the Initialize dataset node. DataWorks can obtain a row of data from the table based on the specified offset. The value of the `${dag.offset}` variable increments from 0. Therefore, the data entries such as `${dag.input[0]}` and `${dag.input[1]}` are returned until all data entries in the dataset are returned.

- iv. Define the condition for exiting the loop for the end node. The values of the `${dag.loopTimes}` and `${dag.input.length}` variables are compared, as shown in the following figure. If the value of the former is less than that of the latter, the end node returns True and the do-while node continues the loop. Otherwise, the end node returns False and the do-while node exits the loop.

```
Language: Python
1 if ${dag.loopTimes}<${dag.input.length}:
2     print True;
3 else
4     print False;
```

Note The system automatically sets the `${dag.input.length}` variable to the number of rows in the array specified by the input parameter based on the context configured for the do-while node.

- 3. Run the do-while node and view the running result.

Summary

- Compared with the while, foreach, and do...while statements, a do-while node has the following characteristics:
 - A do-while node contains a loop body that runs a loop before evaluating the conditional statement. This node functions the same as the do...while statement. A do-while node can also use the system variable `${dag.offset}` and the node context to implement the feature of the foreach statement.
 - A do-while node cannot achieve the feature of the while statement because a do-while node runs a loop before evaluating the conditional statement.
- A do-while node works in the following way:
 - i. Run nodes in the loop body starting from the start node based on node dependencies.
 - ii. Run the code defined for the end node.
 - Run the loop again if the end node returns True.
 - Exit the loop if the end node returns False.
- How to use the node context: The internal nodes of a do-while node can use `${dag.${ctxKey}}` to reference the context defined for the do-while node.
- System parameters: DataWorks provides the following system variables for the internal nodes of the do-

while node:

- $\{\text{dag.loopTimes}\}$: the loop count, starting from 1.
- $\{\text{dag.offset}\}$: the offset of the loop count to 1, starting from 0.

2.5.5.7.3. Create a merge node

This topic describes the definition of merge nodes and how to create a merge node and define the merging logic. It also provides an example to show the scheduling configuration and running details of a merge node.

A merge node is a logical control node in DataStudio. It can merge the running results of its parent nodes, regardless of their running statuses. It aims at facilitating the running of nodes that depend on the output of the child nodes of a branch node.

You cannot change the running status of a merge node. A merge node merges the running results of multiple child nodes of a branch node and sets the running status to Successful. To guarantee the proper running of a node that depends on the output of the child nodes of a branch node, you can configure the node to directly depend on the merge node.

For example, Branch node C has two logically exclusive branches C1 and C2. These two branches use different logic to write data to the same MaxCompute table. Assume that Node B depends on the output of this MaxCompute table. To make sure that Node B can run properly, you must use Merge node J to merge the running results of branches C1 and C2, and then configure Merge node J as the parent node of Node B. If Node B directly depends on branches C1 and C2, one of the branches will fail to run because only one branch meets the branch condition each time Branch node C runs. In this case, Node B cannot be triggered as scheduled.

Create a merge node

1. [Log on to the DataWorks console.](#)
2. On the Data Analytics tab, move the pointer over  and choose **General > MERGE Nodes**.

Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > MERGE Nodes**.

3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive.

4. Click **Commit**.

Define the merging logic

After the merge node is created, the node configuration tab appears. Specify the branches to be merged for the node. Enter the output name or output table name of the parent node, and click the **Add** icon. You can view the running status in the **Result** section. The available running statuses are **Successful** and **Branch Not Running**.

Click the **Properties** tab in the right-side navigation pane and configure the scheduling properties of the merge node.

Run the merge node

If a branch meets the specified condition, the branch is run. You can select the branch and view the running details on the **Runtime Logs** tab.

If a branch does not meet the specified condition, the branch is skipped. You can select the branch and view related information on the **Runtime Logs** tab.

2.5.5.7.4. Create a branch node

A branch node is a logical control node in DataStudio. It can define the branch logic and the direction of branches under different logical conditions.

Prerequisites

Generally, branch nodes need to be used with assignment nodes.

Create a branch node

1. [Log on to the DataWorks console.](#)
2. On the Data Analytics tab, move the pointer over  and choose **General > Branch Node**. Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Branch Node**.
3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive.

4. Click **Commit**.
5. Define the branch logic
 - i. In the **Definition** section, click **Add Branch**.
 - ii. In the **Branch Definition** dialog box, set the parameters.

Parameter	Description
Condition	<p>The condition of the branch.</p> <ul style="list-style-type: none">■ You can only use Python comparison operators to define logical conditions for the branch node.■ If the result of the expression is <i>true</i> when the node is running, the corresponding branch condition is met.■ If the expression fails to be parsed when the node is running, the whole branch node fails.■ To define branch conditions, you can use global variables and parameters defined in the node context. For example, the <code>#{input}</code> variable can be used as an input parameter of the branch node.
Associated Node Output	<p>The associated node output of the branch.</p> <ul style="list-style-type: none">■ The node output is used to configure dependencies for the child nodes of the branch node.■ If the branch condition is met, the child node corresponding to the node output is run. If the child node also depends on the output of other nodes, the status of these nodes is considered.■ If the branch condition is not met, the child node corresponding to the node output is not run. The child node is set to the Not Running state.
Description	<p>The description of the branch. For example, the branches <code>#{input}==1</code> and <code>#{input}>2</code> are defined.</p>

- iii. Click **OK**. After you add a branch, you can click **Change** or **Delete** in the Actions column of the branch to modify or delete it.
 - Click **Change** to modify the branch and related dependencies.
 - Click **Delete** to delete the branch and related dependencies.
6. On the configuration tab of the branch node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, set parameters in the Schedule section. After the branch conditions are defined, the output names are automatically added to the **Outputs** section on the **Properties** tab. Then, you can associate child nodes with the branch node based on the output names.

 **Note**

- Child nodes inherit dry-run properties of the parent node. Therefore, we recommend that you do not create a node depending on its last-cycle instance as the branch.
- The dependencies established by drawing lines between nodes on the dashboard of a workflow are not recorded on the Properties tab. You must manually enter these dependencies.

7. Commit the node.

 **Notice** You must set **Rerun** and **Parent Nodes** before you can commit the node.

- i. Click  in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
- iii. Click **OK**.

In a workspace in standard mode, you need to click **Deploy** in the upper-right corner after you commit the branch node.

8. Test the node.

Supported Python comparison operators

In the following table, assume that the value of the a variable is 10 and that of the b variable is 20.

Comparison operator	Description	Example
==	Equal: checks whether two objects are equal.	(a==b) returns false.
!=	Not equal: checks whether two objects are not equal.	(a!=b) returns true.
<>	Not equal: checks whether two objects are not equal.	(a<>b) returns true. This operator is similar to !=.
>	Greater than: checks whether the variable on the left side of the operator is greater than that on the right side.	(a>b) returns false.

Comparison operator	Description	Example
<	Less than: checks whether the variable on the left side of the operator is less than that on the right side. If the return result is 0 or 1, 0 indicates false and 1 indicates true. These two results are equivalent to the special variables true and false, respectively.	(a<b) returns true.
>=	Greater than or equal to: checks whether the variable on the left side of the operator is greater than or equal to that on the right side.	(a>=b) returns false.
<=	Less than or equal to: checks whether the variable on the left side of the operator is less than or equal to that on the right side.	(a<=b) returns true.

2.5.5.7.5. Create an assignment node

An assignment node uses one of the three value assignment languages MaxCompute SQL, SHELL, and Python to assign values by using the outputs parameter. This node is used to transmit data between a parent node and a child node based on context-based parameters.

Context

The outputs parameter has the following limits:

- The value of the outputs parameter is taken only from the output of the last line of the code.
 - If you use MaxCompute SQL, the output of the SELECT statement in the last line is used.
 - If you use SHELL, the output of the ECHO statement in the last line is used.
 - If you use Python, the output of the PRINT statement in the last line is used.
- The passed value of the outputs parameter is limited to 2 MB in size. If the output of the assignment statement exceeds this limit, the assignment node fails to run.

 **Note** If you use Python or SHELL, the value of the outputs parameter is a one-dimensional array where elements are separated with commas (.). If you use MaxCompute SQL, the value of the outputs parameter is passed to child nodes as a two-dimensional array.

Create an assignment node

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **General > Assignment Node**.
Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Assignment Node**.
3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Notice** A node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.). It is not case-sensitive.

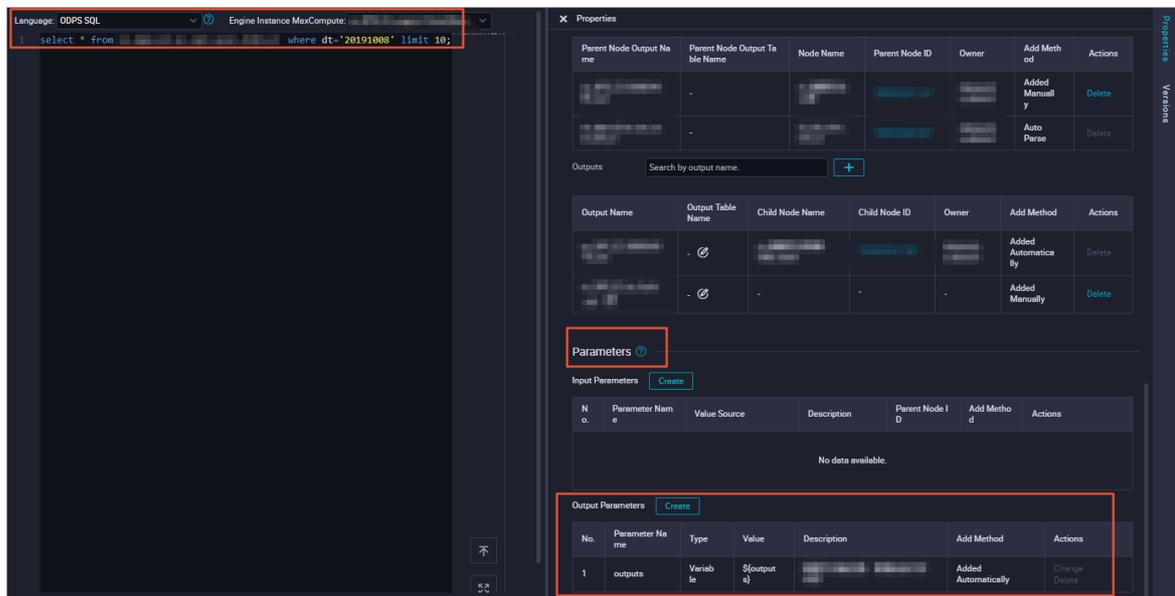
This following sections describe how to use assignment nodes that use the Python, MaxCompute SQL, and SHELL languages respectively to pass data between a parent node and a child node named Assignment node value comparison_shell by using context-based parameters.

After the assignment nodes that use the Python, MaxCompute SQL, and SHELL languages are created, you must set the dependencies so that the child node can reference the parameter values passed by these nodes.

4. Click **Commit**.

Configure the child node to reference the output values of the assignment node that uses MaxCompute SQL

1. Find the target workflow and double-click the assignment node `fuzhi_sql` that uses MaxCompute SQL.
2. On the configuration tab of the `fuzhi_sql` node that appears, click **Properties** in the right-side navigation pane.
3. Configure the `fuzhi_sql` node. The `fuzhi_sql` node assigns the results queried from a specified table to the outputs parameter.



4. Double-click the Assignment node `value comparison_shell` node, which is the child node of the `fuzhi_sql` node.
5. On the configuration tab of the Assignment node `value comparison_shell` node that appears, click **Properties** in the right-side navigation pane and configure the node. The Assignment node `value comparison_shell` node depends on the `fuzhi_sql` node and uses the value of the `outputs` parameter of the `fuzhi_sql` node as the value of its input parameter `sql_inputs`.

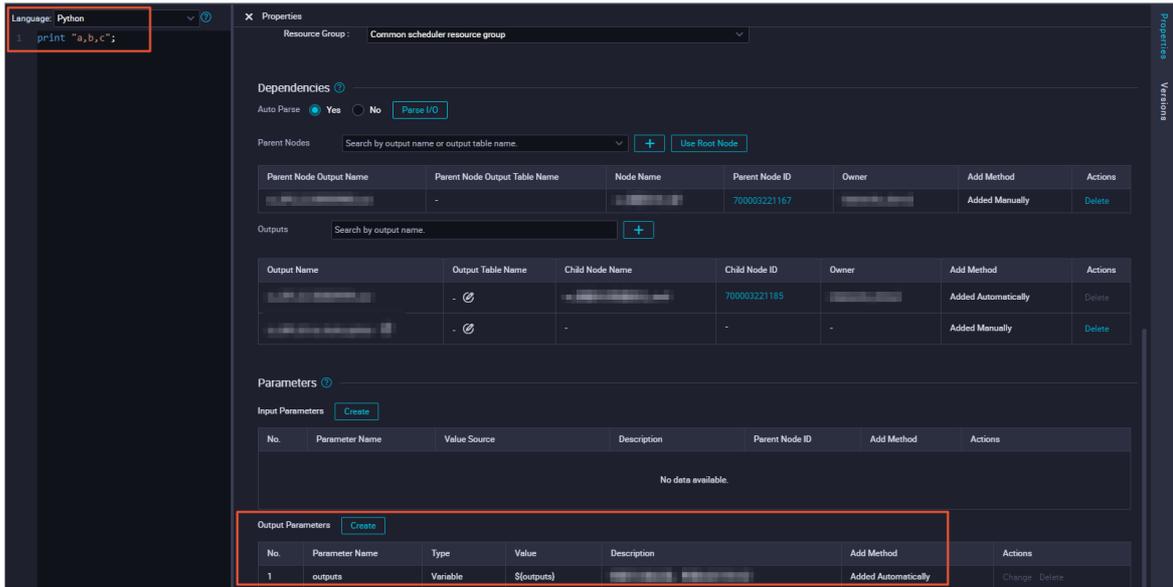
```
echo '${sql_inputs}';
echo 'Use the value in the first line in the output of the fuzhi_sql node as the input'${sql_inputs[0]};
echo 'Use the value in the second line in the output of the fuzhi_sql node as the input'${sql_inputs[1]};
echo 'Use the value of the second field in the first line in the output of the fuzhi_sql node as the input'${sql_inputs[0][1]};
echo 'Use the value of the third field in the second line in the output of the fuzhi_sql node as the input'${sql_inputs[1][2]};
```

6. Click  in the toolbar.
7. In the **Warning** message, click **Continue to Run**.

8. View the result.

Configure the child node to reference the output values of the assignment node that uses Python

1. Find the target workflow and double-click the assignment node `fuzhi_python` that uses Python.
2. On the configuration tab of the `fuzhi_python` node that appears, click **Properties** in the right-side navigation pane.
3. Configure the `fuzhi_python` node. The `fuzhi_python` node assigns the values `a,b,c` to the outputs parameter.



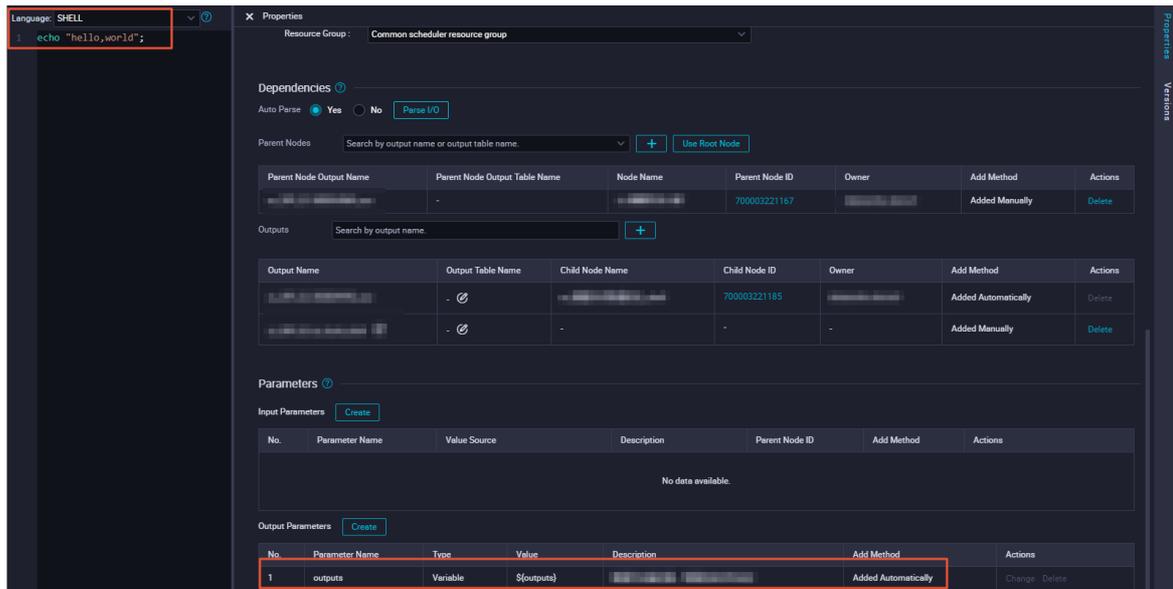
4. Double-click the Assignment node value `comparison_shell` node, which is the child node of the `fuzhi_python` node.
5. On the configuration tab of the Assignment node value `comparison_shell` node that appears, click **Properties** in the right-side navigation pane and configure the node. The Assignment node value `comparison_shell` node depends on the `fuzhi_python` node and uses the value of the outputs parameter of the `fuzhi_python` node as the value of its input parameter `python_inputs`.

```
echo 'The output of the fuzhi_python node'${python_inputs};
echo 'Use the first value in the output of the fuzhi_python node as the input'${python_inputs[0]};
echo 'Use the second value in the output of the fuzhi_python node as the input'${python_inputs[1]};[1]
```

6. Click  in the toolbar.
7. In the **Warning** message, click **Continue to Run**.
8. View the result.

Configure the child node to reference the output values of the assignment node that uses SHELL

1. Find the target workflow and double-click the assignment node `fuzhi_shell` that uses SHELL.
2. On the configuration tab of the `fuzhi_shell` node that appears, click **Properties** in the right-side navigation pane.
3. Configure the `fuzhi_shell` node. The `fuzhi_shell` node assigns the values `hello,world` to the outputs parameter.



4. Double-click the Assignment node value comparison_shell node, which is the child node of the fuzhi_shell node.
5. On the configuration tab of the Assignment node value comparison_shell node that appears, click **Properties** in the right-side navigation pane and configure the node. The Assignment node value comparison_shell node depends on the fuzhi_shell node and uses the value of the outputs parameter of the fuzhi_shell node as the value of its input parameter shell_inputs.

```
echo 'The output of the fuzhi_shell node'${shell_inputs};
echo 'Use the first value in the output of the fuzhi_shell node as the input'${shell_inputs[0]};
echo 'Use the second value in the output of the fuzhi_shell node as the input'${shell_inputs[1]};
```

6. Click  in the toolbar.
7. In the **Warning** message, click **Continue to Run**.
8. View the result.

2.5.5.7.6. Create a Shell node

Shell nodes support standard shell syntax but not interactive syntax.

Procedure

1. [Log on to the DataWorks console](#).
2. On the Data Analytics tab, move the pointer over  and choose **General > Shell**. Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Shell**.
3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.). It is not case-sensitive.

4. Click **Commit**.
5. Edit the Shell node.

- i. Edit the code on the configuration tab of the Shell node. To call the system scheduling parameters for the Shell node, execute the following statement:

```
echo "$1 $2 $3"
```

 **Note** Separate multiple parameters with spaces.

- i. Click  in the toolbar to save the SQL statement to the server.
 - ii. Click  in the toolbar to execute the SQL statement you have saved. If you need to change the resource group used to test the Shell node on the **DataStudio** page, click  in the toolbar and select your desired exclusive resource group.
6. On the configuration tab of the Shell node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, set parameters in the Schedule section.
 7. Commit the node.

 **Notice** You must set **Rerun** and **Parent Nodes** before you can commit the node.

- i. Click  in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
- iii. Click **OK**.

In a workspace in standard mode, you need to click **Deploy** in the upper-right corner after you commit the branch node.

8. Test the node.

2.5.5.7.7. Create a zero-load node

A zero-load node is a control node, which only supports dry-run scheduling and does not generate any data. It usually serves as the root node of a workflow.

Context

You can configure an output table for a zero-load node so that the output table can be used as an input table of another node. However, the zero-load node does not process the table data.

Procedure

1. [Log on to the DataWorks console.](#)
2. On the **Data Analytics** tab, move the pointer over  and choose **General > Zero-Load Node**.
Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Zero-Load Node**.
3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive.

4. Click **Commit**.
5. On the configuration tab of the zero-load node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, set parameters in the Schedule section. For more information, see [Basic](#)

properties.

6. Commit the node.

 **Notice** You must set **Rerun** and **Parent Nodes** before you can commit the node.

- i. Click  in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
- iii. Click **OK**.

In a workspace in standard mode, you need to click **Deploy** in the upper-right corner after you commit the branch node.

7. Test the node.

2.5.5.7.8. Create a cross-tenant collaboration node

Cross-tenant collaboration nodes are used to associate nodes from different tenants. Cross-tenant collaboration nodes are classified into sender nodes and receiver nodes.

Prerequisites

A sender node and its receiver node use the same CRON expression. You can click the **Properties** tab in the right-side navigation pane of a node configuration tab and view the CRON expression in the **Schedule** section.

Create a cross-tenant collaboration node

1. [Log on to the DataWorks console](#).
2. On the Data Analytics tab, move the pointer over  and choose **General > Cross-Tenant Collaboration**.

Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Cross-Tenant Collaboration**.

3. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive.

4. On the node configuration tab, set the parameters in the **Cross-Tenant Collaboration** section.

Parameter	Description
Type	The type of the cross-tenant collaboration node. Valid values: Sender and Receiver .
Location	The path of the cross-tenant collaboration node. The node path cannot be modified.
Collaborative Workspaces	The workspace name and Apsara Stack tenant account of the peer node. This example sets the node type to Sender . Therefore, you must enter the workspace name and Apsara Stack tenant account of the receiver node.

5. After the sender node is created, follow the same procedure to create the receiver node under the Apsara Stack tenant account and workspace to which the receiver node belongs.

Set the node type to **Receiver**. The information about available sender nodes appears. You must also set **Timeout**. This parameter indicates the timeout period of the receiver node after it starts running.

The sender node first sends a message to the message center. After the message is delivered, the status of the sender node is set to successful. The receiver node continuously pulls messages from the message center. If a message is received within the timeout period, the status of the receiver node is set to successful.

If the receiver node does not receive any messages within the timeout period, the receiver node fails. The lifecycle of a message is 24 hours.

Assume that an auto triggered instance was run on October 8, 2018. A message indicating the completion of the instance was then sent to the message center. If you create a retroactive instance for the receiver node with the data timestamp set to October 7, 2018, the status of the generated receiver node instance is set to successful.

6. After the configuration is completed, save and commit the node.

2.5.5.7.9. Create a data analysis report node

A data analysis report node is used to associate a report in the DataAnalysis module with the parent nodes on which the report depends and update the report as scheduled.

Prerequisites

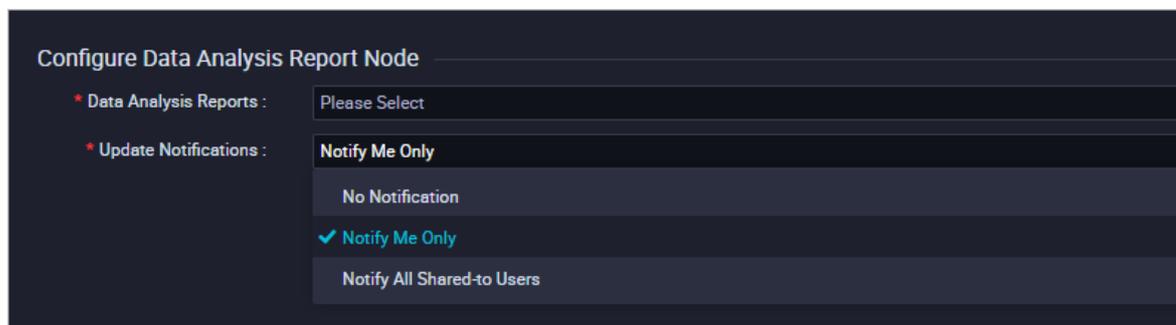
A table is created on the **Report** page of the **DataAnalysis** module.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **General > Data Analysis Reports**.
Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Data Analysis Reports**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab, set the parameters in the **Configure Data Analysis Report Node** section.



Parameter	Description
-----------	-------------

Parameter	Description
Data Analysis Reports	The report for which you want to receive the notifications about the updates. Select a report created on the Report page of the DataAnalysis module.
Update Notifications	Specifies the users who can receive the notifications when the report is updated. Valid values: No Notification , Notify Me Only , and Notify All Shared to Users .

6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the data analysis report node. For more information, see [Recurring tasks](#). After the node is run in the production environment, you can view updates of the report on the Report page of the **DataAnalysis** module.

2.5.5.7.10. Create a real-time sync node check node

This topic describes how to create and configure a real-time sync node check node.

Context

DataWorks provides the real-time sync node check node for you to associate the status of the current node with scheduling of its child node, which is a batch sync node. After you set dependencies in the workflow, the scheduling of the child node is determined by the status of the current node. This node allows you to schedule real-time sync nodes and batch sync nodes in a unified manner.

Create a real-time sync node check node

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **General > Check Realtime Compute Node**. Alternatively, you can click a workflow in the Business Flow section, right-click **General**, and then choose **Create > Check Realtime Compute Node**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

4. Click **Commit**.
5. On the node configuration tab that appears, set the parameters in the **Check Realtime Compute Node** section.

Parameter	Description
StreamCompute Node	<p>The real-time sync node in Stream Studio.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note The node must be deployed in Stream Studio.</p> </div>
Control rule	<p>The control rule that determines whether to schedule or block the child node based on the status of the current node.</p> <ul style="list-style-type: none"> ○ The status of the current node can be Normal or Paused. <ul style="list-style-type: none"> ■ Normal: The node is running and no error is reported. ■ Paused: The node is stopped or paused manually, or the node stops or cannot be restarted due to an exception. ○ The control method for the child node can be Scheduled or Blocked. <ul style="list-style-type: none"> ■ Scheduled: The child node can be run as scheduled. ■ Blocked: The child node cannot be scheduled, regardless of the settings.

6. Commit the node.

- i. Click  in the toolbar.
- ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
- iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the real-time sync node check node. For more information, see [Publish nodes](#).

2.5.5.8. Custom

2.5.5.8.1. Create a Hologres development node

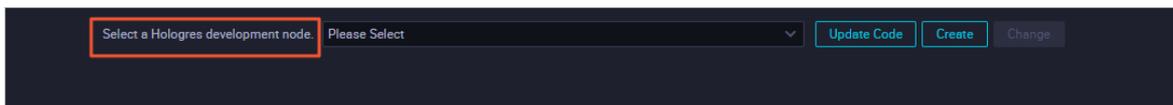
This topic describes how to create and modify a Hologres development node and update the node version.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, move the pointer over  and choose **Custom > Hologres Development**.
Alternatively, you can click a workflow in the Business Flow section, right-click **UserDefined**, and then choose **Create > Hologres Development**.
3. In the **Create Node** dialog box, specify **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`).

4. Click **Commit**.
5. On the node configuration tab that appears, select a Hologres development node.



If no Hologres node is available, click **Create** to create one. You can also click **Change** to modify an existing node.

6. On the configuration tab of the batch synchronization node, click the **Properties** tab in the right-side navigation pane. On the Properties tab, configure parameters in the Schedule section. For more information, see [Basic properties](#).
7. Commit the node.
 - i. Click  in the toolbar.
 - ii. In the **Commit Node** dialog box, enter your comments in the **Description** field.
 - iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see [Publish nodes](#).

8. Test the node. For more information, see [Recurring tasks](#).

2.5.6. Schedule

2.5.6.1. Basic properties

On the Properties tab of a node, you can set parameters of the node in the General, Schedule, Dependencies, and Parameters sections. The General section allows you to set the basic properties of the node.

On the **Data Analytics** tab of the DataStudio page, double-click a node. On the node configuration tab that appears, click the **Properties** tab in the right-side navigation pane and set the parameters in the **General** section.

Parameter	Description
Node Name	The name of the node that you set when creating the node. To modify the name, right-click the node in the left-side navigation pane and select Rename .
Node ID	The unique ID of the node. The node ID is generated when the node is committed at the first time. The node ID cannot be modified.
Node Type	The type of the node that you set when creating the node. The node type cannot be modified.
Owner	<p>The owner of the node. By default, the owner of a newly created node is the current logon user. You can change the owner.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note Only a member in the workspace where the node resides can be selected as the owner.</p> </div>
Description	The description of the node, such as the business and usage.
Arguments	The parameter used to assign a value to a variable in the code during node scheduling. You can enter multiple parameters. Separate multiple parameters with spaces.

Parameter value assignment formats for various node types

- Format for ODPS SQL and ODPS MR nodes: `Variable name 1=Parameter 1 Variable name 2=Parameter 2 .`
Separate multiple parameters with spaces.

- Format for Shell nodes: `Parameter 1 Parameter 2` . Separate multiple parameters with spaces.

For more information about the built-in scheduling parameters, see [Parameter configuration](#).

2.5.6.2. Scheduling parameters

In common data development scenarios, the code of different types of nodes may be subject to change from time to time. You must dynamically modify the values of some parameters, such as the date and time, based on the requirement changes and time changes.

In this case, you can use the scheduling parameter configuration feature of DataWorks. After relevant parameters are set, auto triggered nodes can automatically parse the code to obtain required data. Configurable parameters in DataWorks are classified into system parameters and custom parameters. We recommend that you use custom parameters.

```
{
  "data":[
    {
      "beginRunningTime":"1564019679966",
      "beginWaitResTime":"1564019679966",
      "beginWaitTimeTime":"1564019679506",
      "bizdate":"1559318400000",
      "createTime":"1564019679464",
      "dagId":332455685,
      "dagType":5,
      "finishTime":"1564019679966",
      "instanceId":2427622331,
      "modifyTime":"1564019679966",
      "nodeName":"vi","status":6
    }
  ],
  "errCode":"0",
  "errMsg": "",
  "requestId":"E17535-8C06-43F6-B1EA-6236FE9",
  "success":true
}
```

Auto-completion is supported when you specify a data type for a parameter.

Parameter types

Parameter type	Configuration method	Applicable to	Example
----------------	----------------------	---------------	---------

Parameter type	Configuration method	Applicable to	Example
System parameters: including bdp.system.bizdate and bdp.system.cyctime	To use the system parameters in the scheduling system, reference <code>\${bdp.system.bizdate}</code> and <code>\${bdp.system.cyctime}</code> in the code, instead of setting them in the Arguments field. The system can automatically replace the values of the parameters that reference the system parameters in the code.	All nodes	N/A
Non-system parameters: custom parameters (recommended)	Reference <code>#{key1}</code> and <code>#{key2}</code> in the code and set them in the Arguments field, for example, <code>"key1=value1 key2=value2"</code> .	Non-Shell nodes	<ul style="list-style-type: none"> • Constant parameters: <code>param1="abc"param2=1234</code>. • Variables: <code>param1=\${yyyyymmdd}</code>, the value of which is calculated based on the value of <code>bdp.system.cyctime</code>.
	Reference <code>\$1</code> , <code>\$2</code> , and <code>\$3</code> in the code and set them in the Arguments field, for example, <code>"value1 value2 value3"</code> .	Shell nodes	<ul style="list-style-type: none"> • Constant parameters: <code>"abc" 1234</code>. • Variables: <code>#{yyyyymmdd}</code>, the value of which is calculated based on the value of <code>bdp.system.cyctime</code>.

As described in the preceding table, the values of custom variables are calculated based on the values of system parameters. You can use custom variables to flexibly define the data to be obtained and the data format. For custom parameters, the following types of brackets are used:

- Braces { } define the data timestamp. For example, the value of `{yyyyymmdd}` is calculated based on the value of `bdp.system.bizdate`.
- Brackets [] define the running time. For example, the value of `[yyyyymmddhh]` is calculated based on the value of `bdp.system.cyctime`.

Note Nodes can be scheduled only in the production environment. Therefore, the values of scheduling variables are replaced only after nodes are run in the production environment.

After you set the scheduling variables for a node, you can click the **Run Smoke Test in Development Environment** icon on the node configuration tab to test whether the values of scheduling variables can be replaced as expected during node scheduling.

You can click the **Properties** tab in the right-side navigation pane, and assign values to scheduling variables in the **Arguments** field in the **General** section. Note the following issues when you set parameters:

- Do not add spaces on either side of the equal sign (=) for a parameter. For example, enter `bizdate=$bizdate`.
- Separate multiple parameters (if any) with spaces. For example, enter `bizdate=$bizdate datetime=${yyyymmdd}`.

System parameters

DataWorks provides the following system parameters:

- `${bdp.system.cyctime}`: the scheduled time to run an instance. Default format: `yyyymmddhh24miss`. This parameter can specify the hour and minutes of the scheduled time.
- `${bdp.system.bizdate}`: the timestamp of data to be analyzed by an instance. Default format: `yyyymmdd`. The default data timestamp is one day before the scheduled time.

Use the following formula to calculate the running time based on the data timestamp: `Running time = Data timestamp + 1`.

To use the system parameters, you can reference them in the code, instead of setting them in the **Arguments** field. The system can automatically replace the values of the parameters that reference the system parameters in the code.

Note The scheduling properties of an auto triggered node are configured to define the scheduling rules of the running time. Therefore, you can calculate the data timestamp based on the scheduled time to run an instance and obtain the values of system parameters for the instance.

Example of system parameters

For example, to set an ODPS SQL node to run once per hour from 00:00 to 23:59 every day, perform the following steps if you want to use system parameters in the code:

1. Reference system parameters in the code.

```
insert overwrite table tb1 partition(ds='20150304') select
c1,c2,c3
from (
select * from tb2
where ds='${bdp.system.cyctime}') t
full outer join(
select * from tb3
where ds='${bdp.system.bizdate}') y
on t.c1=y.c1;
```

2. After the preceding step, your node is partitioned by using the system parameters. Set the scheduling properties and dependencies. For more information, see [Schedule](#) and [Dependencies](#). In this example, the node is scheduled by hour.
3. After you set the recurrence and dependencies, commit and deploy the node. Then, you can check the

node in **Manage auto triggered nodes**. The scheduling system generates instances for the auto triggered node from the second day. You can right-click an instance in the directed acyclic graph (DAG) and select **View Runtime Log** to view the parsed values of the system parameters.

For example, the scheduling system generated 24 running instances for the node on January 14, 2019. The data timestamp is January 13, 2019 for all instances. Therefore, the value of `bdp.system.bizdate` is 20190113. The running time is the running date appended with the scheduled time. Therefore, the value of `bdp.system.cyctime` is 20190114000000 plus the scheduled time of each instance.

Open the runtime logs of each instance and search for the replaced values of the system parameters in the code:

- The scheduled time for the first instance is January 14, 2019 00:00:00. Therefore, `bdp.system.bizdate` is replaced with 20190113 and `bdp.system.cyctime` is replaced with 20190114000000.
- The scheduled time for the second instance is January 14, 2019 01:00:00. Therefore, `bdp.system.bizdate` is replaced with 20190113 and `bdp.system.cyctime` is replaced with 20190114010000.
- Similarly, the scheduled time for the twenty-fourth instance is January 14, 2019 23:00:00. Therefore, `bdp.system.bizdate` is replaced with 20190113 and `bdp.system.cyctime` is replaced with 20190114230000.

Custom parameters for non-Shell nodes

To set scheduling variables for a non-Shell node, add `#{Variable name}` in the code to reference the function and assign a value to the scheduling variable.

Note The name of a variable in the SQL code can contain only letters, digits, and underscores (`_`). If the variable name is `date`, the value of `bizdate` is automatically assigned to this variable. For more information, see the "Built-in scheduling parameters" section in this topic. You do not need to assign a value in the Arguments field. Even if another value is assigned, it is not used in the code because the value of `bizdate` is automatically assigned in the code.

Example of custom parameters for non-Shell nodes

For example, to set an ODPS SQL node to run once per hour from 00:00 to 23:59 every day, perform the following steps if you want to use the hour-related custom variables `thishour` and `lasthour` in the code:

1. Reference the parameters in the code.

```
insert overwrite table tb1 partition(ds='20150304') select
c1,c2,c3
from (
select * from tb2
where ds='#{thishour}') t
full outer join(
select * from tb3
where ds = '#{lasthour}') y
on t.c1=y.c1;
```

2. Click the **Properties** tab in the right-side navigation pane of the node configuration tab. Assign values to the custom parameters referenced in the code in the **Arguments** field in the **General** section.

Set the custom parameters in the following formats:

- `thishour=${yyyy-mm-dd/hh24:mi:ss}`
- `lasthour=${yyyy-mm-dd/hh24:mi:ss-1/24}`

 **Note** The value of yyyy-mm-dd/hh24:mi:ss corresponds to that of cyctime. For more information, see the "Custom parameters" section in this topic.

You can enter `thishour=${yyyy-mm-dd/hh24:mi:ss} lasthour=${yyyy-mm-dd/hh24:mi:ss-1/24}` in the **Arguments** field.

3. Set the node to run once per hour.
4. After you set the recurrence and dependencies, commit and deploy the node. Then, you can check the node in **Operation Center**. The scheduling system generates instances for the auto triggered node from the second day. You can right-click an instance in the DAG and select **View Runtime Log** to view the parsed values of the custom parameters. The value of cyctime is 20190114010000. Therefore, the value of thishour is 2019-01-14/01:00:00 and the value of lasthour, which indicates the last hour, is 2019-01-14/00:00:00.

Custom parameters for Shell nodes

The parameter configuration procedure of a Shell node is similar to that of a non-Shell node, except that the variable naming rules are different. Variable names for a Shell node cannot be customized, but must follow the \$1,\$2,\$3... format. For example, add \$1 in the code of a Shell node and enter the built-in scheduling parameter \$xxx in the **Arguments** field. Then, the value of \$xxx can replace that of \$1 in the code.

 **Note** If the number of parameters in a Shell node reaches 10, use \${10} to declare the tenth variable.

Example of custom parameters for Shell nodes

For example, set a Shell node to run at 01:00 every day. To use the custom constant parameter myname and the custom variable ct in the code, perform the following steps:

1. Reference the parameters in the code.

```
echo "hello $1, two days ago is $2, the system param is ${bdp.system.cyctime}";
```

2. Click the **Properties** tab in the right-side navigation pane of the node configuration tab. Assign values to the custom parameters referenced in the code in the **Arguments** field in the **General** section. Separate multiple parameters with spaces, for example, enter Parameter 1 Parameter 2 Parameter 3. The custom parameters are parsed based on the parameter sequence. For example, \$1 is replaced with the value of Parameter 1. In this example, enter abcd \${yyyy-mm-dd-2} in the **Arguments** field to set \$1 and \$2 to abcd and \${yyyy-mm-dd-2}, respectively.
3. Set the node to run at 01:00 every day.
4. After you set the recurrence and dependencies, commit and deploy the node. Then, you can check the node in **Operation Center**. The scheduling system generates instances for the auto triggered node from the second day. Right-click an instance in the DAG and select **View Runtime Log**. The logs show that \$1 in the code is replaced with abcd, \$2 is replaced with 2019-01-12 (two days before the running date), and \${bdp.system.cyctime} is replaced with 20190114010000.

Custom parameters

Custom parameters are divided into constant parameters and variables based on the value type. DataWorks provides some built-in scheduling parameters as variables.

- Constant parameters

For example, for an SQL node, add \${Variable name} in the code and set the following parameter for the node: Variable name=Fixed value.

- Code: `select xxxxxx type='${type}'`

- Value assigned to the scheduling variable: type='aaa'. When the node is run, the variable in the code is replaced with type='aaa'.

- Variables

Variables are built-in scheduling parameters whose values depend on the system parameters `#{bdp.system.bizdate}` and `#{bdp.system.cycetime}`.

For example, for an SQL node, add `#{Variable name}` in the code and set the following parameter for the node: Variable name=Scheduling parameter.

- Code: `select xxxxxx dt=#{datetime}`
- Value assigned to the scheduling variable: `datetime=$bizdate`

If the node is run on July 22, 2017, the variable in the code is replaced with `dt=20170721`.

Built-in scheduling parameters

- `$bizdate`

- Parameter description: the data timestamp in the format of `yyyymmdd`. By default, the value of this parameter is one day before the scheduled time to run a node.
- For example, the code of an ODPS SQL node includes `pt=#{datetime}`, and the parameter configured for the node is `datetime=$bizdate`. If the node is run on July 22, 2017, `$bizdate` is replaced with `pt=20170721`.

- `$cycetime`

- Parameter description: the scheduled time to run a node. If no scheduled time is configured for a node scheduled by day, `$cycetime` is set to 00:00 of the day. The time is accurate to seconds. This parameter is usually used for nodes scheduled by hour or minute.

Note

- Pay attention to the difference between the time parameters configured by using `#{}` and `#{}`. `$bizdate` specifies the data timestamp, which is one day before the current day by default.
- `$cycetime` specifies the scheduled time to run a node. If no scheduled time is configured for a node scheduled by day, `$cycetime` is set to 00:00 of the day. The time is accurate to seconds. This parameter is usually used for nodes scheduled by hour or minute.

For example, if a node is scheduled to run at 00:30 on the current day, `$cycetime` is set to `yyyy-mm-dd 00:30:00`.

- If a time parameter is configured by using `#{}`, `$bizdate` is used as the benchmark for running nodes. The time parameter is replaced with the data timestamp selected for retroactive data generation.
- If a time parameter is configured by using `#{}`, `$cycetime` is used as the benchmark for running nodes. The time is calculated in the same way as the time in Oracle. The time parameter is replaced with the data timestamp selected for retroactive data generation plus one day.

For example, if the data timestamp is set to 20140510 for retroactive data generation, `$cycetime` is replaced with 20140511.

- The following examples show the values of custom parameters when `$cycetime` is set to 20140515103000:
 - `${yyyy} = 2014`, `${yy} = 14`, `${mm} = 05`, `${dd} = 15`, `${yyyy-mm-dd} = 2014-05-15`, `${hh24:mi:ss} = 10:30:00`, `${yyyy-mm-dd hh24:mi:ss} = 2014-05-15 10:30:00`
 - `${hh24:mi:ss - 1/24} = 09:30:00`
 - `${yyyy-mm-dd hh24:mi:ss - 1/24/60} = 2014-05-15 10:29:00`
 - `${yyyy-mm-dd hh24:mi:ss - 1/24} = 2014-05-15 09:30:00`
 - `${add_months(yyyymmdd,-1)} = 20140415`
 - `${add_months(yyyymmdd,-12*1)} = 20130515`
 - `${hh24} = 10`
 - `${mi} = 30`

- Method for testing the `$cycetime` parameter:

After an instance starts to run, right-click the instance in the DAG and select **More**. Check whether the scheduled time is the time at which the instance is run.

- `$jobid`
 - Parameter description: the ID of the workflow to which a node belongs.
 - Example: `jobid=$jobid`.
- `$nodeid`
 - Parameter description: the ID of a node.
 - Example: `nodeid=$nodeid`.
- `$taskid`
 - Parameter description: the instance ID of a node.
 - Example: `taskid=$taskid`.
- `$bizmonth`
 - Parameter description: the month of the data timestamp in the format of `yyyymm`. If the month of a data timestamp is the current month, the value of `$bizmonth` is the month of the data timestamp minus 1. Otherwise, the value of `$bizmonth` is the month of the data timestamp.
 - For example, the code of an ODPS SQL node includes `pt=${datetime}`, and the parameter configured for the node is `datetime=$bizmonth`.

Assume that the current day is July 22, 2017. If the node is run on July 22, 2017, `$bizmonth` is replaced with `pt=201706`.
- `${...}` custom parameters
 - You can customize a time format based on the value of `$bizdate`, where `yyyy` indicates the four-digit year, `yy` indicates the two-digit year, `mm` indicates the month, and `dd` indicates the day. You can use any combination of these parameters, for example, `${yyyy}`, `${yyyymm}`, `${yyyymmdd}`, and `${yyyy-mm-dd}`.
 - `$bizdate` is accurate to the day. Therefore, `${...}` can specify only the year, month, or day.

- The following table describes how to specify other intervals based on \$bizdate.

Interval	Expression
N years later	`\${yyyy+N}`
N years before	`\${yyyy-N}`
N months later	`\${yyyymm+N}`
N months before	`\${yyyymm-N}`
N weeks later	`\${yyyymmdd+7*N}`
N weeks before	`\${yyyymmdd-7*N}`
N days later	`\${yyyymmdd+N}`
N days before	`\${yyyymmdd-N}`

- \$gmtdate
 - Parameter description: the current date in the format of yyyymmdd. By default, the value of this parameter is the current date. During retroactive data generation, the input value is the data timestamp plus one day.
 - For example, the code of an ODPS SQL node includes pt=\${datetime}, and the parameter configured for the node is datetime=\$gmtdate. Assume that the current day is July 22, 2017. If the node is run on July 22, 2017, \$gmtdate is replaced with pt=20170722.
- \${yyyymmdd}
 - Parameter description: the data timestamp in the format of yyyymmdd. The value of this parameter is the same as that of \$bizdate. This parameter supports delimiters, for example, yyyy-mm-dd.
By default, the value of this parameter is one day before the scheduled time to run a node. You can customize a time format for this parameter, for example, yyyy-mm-dd for \${yyyy-mm-dd}.
 - Examples:
 - The code of an ODPS SQL node includes pt=\${datetime}, and the parameter configured for the node is datetime=\${yyyy-mm-dd}. If the node is run on July 22, 2018, \${yyyy-mm-dd} is replaced with pt=2018-07-21.
 - The code of an ODPS SQL node includes pt=\${datetime}, and the parameter configured for the node is datetime=\${yyyymmdd-2}. If the node is run on July 22, 2018, \${yyyymmdd-2} is replaced with pt=20180719.
 - The code of an ODPS SQL node includes pt=\${datetime}, and the parameter configured for the node is datetime=\${yyyymm-2}. If the node is run on July 22, 2018, \${yyyymm-2} is replaced with pt=201805.
 - The code of an ODPS SQL node includes pt=\${datetime}, and the parameter configured for the node is datetime=\${yyyy-2}. If the node is run on July 22, 2018, \${yyyy-2} is replaced with pt=2016.
 - You can assign values to multiple parameters when configuring an ODPS SQL node. For example, set startdatetime=\$bizdate enddatetime=\${yyyymmdd+1} starttime=\${yyyy-mm-dd} endtime=\${yyyy-mm-dd+1}.

FAQ

- Q: The table partition format is pt=yyyy-mm-dd hh24:mi:ss, but spaces are not allowed in scheduling parameters. How can I configure the format of \${yyyy-mm-dd hh24:mi:ss}?

A: Use the custom variables `datetime=${yyyy-mm-dd}` and `hour=${hh24:mi:ss}` to obtain the date and time. Then, join them together to form `pt="${datetime} ${hour}"` in the code. Separate the two variables with a space.

- Q: The table partition is `pt="${datetime} ${hour}"` in the code. To obtain the data for the last hour when the node is run, the custom variables `datetime=${yyyymmdd}` and `hour=${hh24-1/24}` can be used to obtain the date and time, respectively. However, for an instance running at 00:00, it analyzes data for 23:00 of the current day, instead of 23:00 of the previous day. What measures can I take in this case?

A: Modify the formula of `datetime` to `${yyyymmdd-1/24}` and keep the formula of `hour` unchanged at `${hh24-1/24}`. Then, the node can be run properly.

- For an instance that is scheduled to run at 2015-10-27 00:00:00, the values of `${yyyymmdd-1/24}` and `${hh24-1/24}` are 20151026 and 23, respectively. This is because the scheduled time minus 1 hour is a time value that belongs to yesterday.
- For an instance that is scheduled to run at 2015-10-27 01:00:00, the values of `${yyyymmdd-1/24}` and `${hh24-1/24}` are 20151027 and 00, respectively. This is because the scheduled time minus 1 hour is a time value that belongs to the current day.

DataWorks offers the following node running modes:

- Manually run a node in DataStudio: You must assign temporary values to parameters in the code to ensure proper running of the node. The assigned values are not saved as node properties and do not take effect in other node running modes.
- Automatically run a node at specified intervals: No configuration is needed in the Arguments field. The scheduling system automatically replaces the values of parameters based on the scheduled time of the current instance.
- Test a node or generate retroactive data: You must specify the data timestamp. The scheduled time of each instance can be calculated based on the formula described earlier in this topic.

2.5.6.3. Scheduling properties

This topic describes how to configure the scheduling properties of a node, including the recurrence and dependencies.

You can click the **Properties** tab in the right-side navigation pane of the node configuration tab and set the parameters in the **Schedule** section.

Node status

- **Normal**: If you select this option, the node is run based on the recurrence. By default, this option is selected for a node.
- **Dry Run**: If you select this option, the node is run based on the recurrence. However, the scheduling system does not actually run the code but directly returns a success response.
- **Retry Upon Error**: If you select this check box, the node is rerun when it encounters an error. By default, a node can be automatically rerun for a maximum of three times at an interval of 2 minutes.
- **Skip Execution**: If you select this check box, the node is run based on the recurrence. However, the scheduling system does not actually run the code but directly returns a failure response. You can select this check box if you want to suspend a node and run it later.

Recurrence

After a node is committed and deployed, the scheduling system generates instances every day from the next day based on the scheduling properties of the node. Then, the scheduling system runs the instances based on the running results of ancestor instances and the scheduled time. If a node is committed and deployed after 23:30, the scheduling system generates instances for it from the third day.

Note

If you schedule a node to run every Monday, the node is run only on Mondays. On the other days, the scheduling system does not actually run the code but directly returns a success response. When you test a node scheduled by week or generate retroactive data for the node, you must set the data timestamp to one day earlier than the scheduled time to run the node.

For an auto triggered node, its dependencies take priority over other scheduling properties. That is, when the scheduled time arrives, the scheduling system does not immediately run a node instance but first checks whether all the ancestor instances are run.

- The node instance is in the Not Running state if any ancestor instances are not run when the scheduled time arrives.
- The node instance is in the Pending (Schedule) state if the scheduled time does not arrive but all the ancestor instances are run.
- The node instance is in the Pending (Resources) state if all the ancestor instances are run and the scheduled time arrives.

Cross-cycle dependencies

DataWorks supports the following three types of cross-cycle dependencies:

- Dependency on instances of child nodes
 - Node dependency: The current node depends on the last-cycle instances of its child nodes. For example, Node A has three child nodes B, C, and D. If you select this node dependency, Node A depends on the last-cycle instances of nodes B, C, and D.
 - Business scenario: The current node depends on instances of child nodes in the last cycle to cleanse the output tables of the current node and check whether the final result is generated properly.
- Dependency on instances of the current node
 - Node dependency: The current node depends on its last-cycle instances.
 - Business scenario: The current node depends on the data output result of its last-cycle instances.
- Dependency on instances of custom nodes: If you select this node dependency, enter the IDs of the nodes on which the current node depends. You can specify multiple nodes and separate their IDs with commas (,). For example, enter 12345,23456.
 - Node dependency: The current node depends on the last-cycle instances of custom nodes.
 - Business scenario: In the business logic, the current node depends on the proper output of other business data that is not processed by the current node.

Note The difference between cross-cycle dependencies and dependencies in the current cycle lies in that cross-cycle dependencies are displayed as dotted lines in Operation Center.

Before deleting a node from Operation Center, you must delete all dependencies of the node so that other nodes can run properly.

Scheduled by day

Nodes scheduled by day are automatically run once per day. When you create an auto triggered node, the node is set to run at 00:00 every day by default. You can specify another time as needed. In the example shown in the following figure, the time is specified as 13:00.

- If you select Customize Runtime, the node is run at the specified time every day. The time format is YYYY-MM-DD HH:MM:SS.

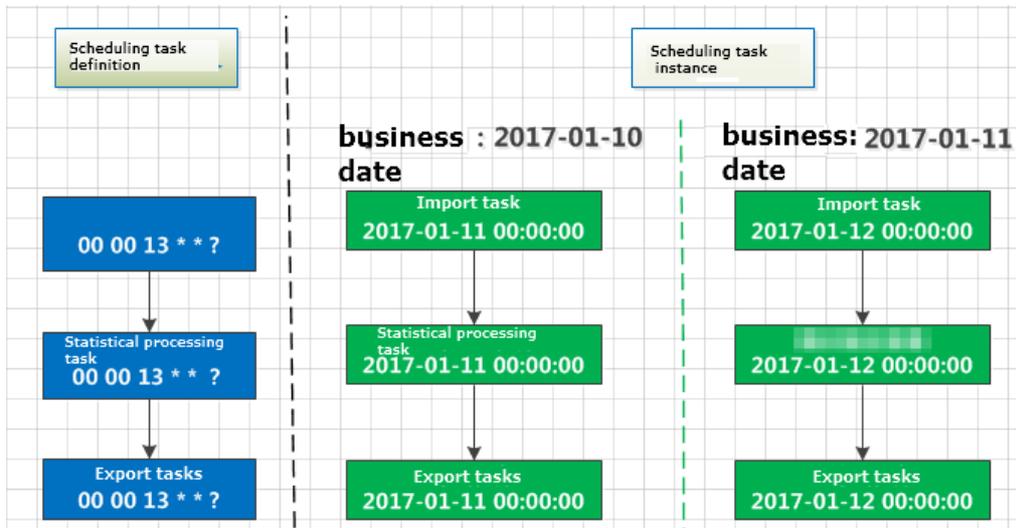
Note An auto triggered node can be run only when all the ancestor instances are run and the scheduled time arrives. Both prerequisites are indispensable and have no specific chronological order.

- If you clear Customize Runtime, the scheduled time of the node is randomly set in the range of 00:00 to 00:30.

Scenarios:

For example, you have created an import node, an analytics node, and an export node. They are all scheduled to run at 13:00 every day. The analytics node depends on the import node, and the export node depends on the analytics node. The following figure shows that the analytics node is configured to depend on the import node.

Based on the preceding node scheduling properties, the scheduling system automatically generates and runs instances for the nodes, as shown in the following figure.

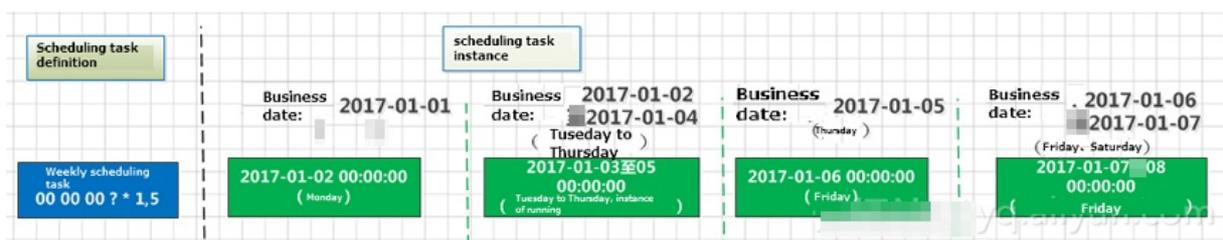


Scheduled by week

Nodes scheduled by week are automatically run at a specified time of specified days every week. On the other days, the scheduling system still generates instances to make sure the proper running of descendant instances. However, the system does not actually run the code or consume resources but directly returns a success response.

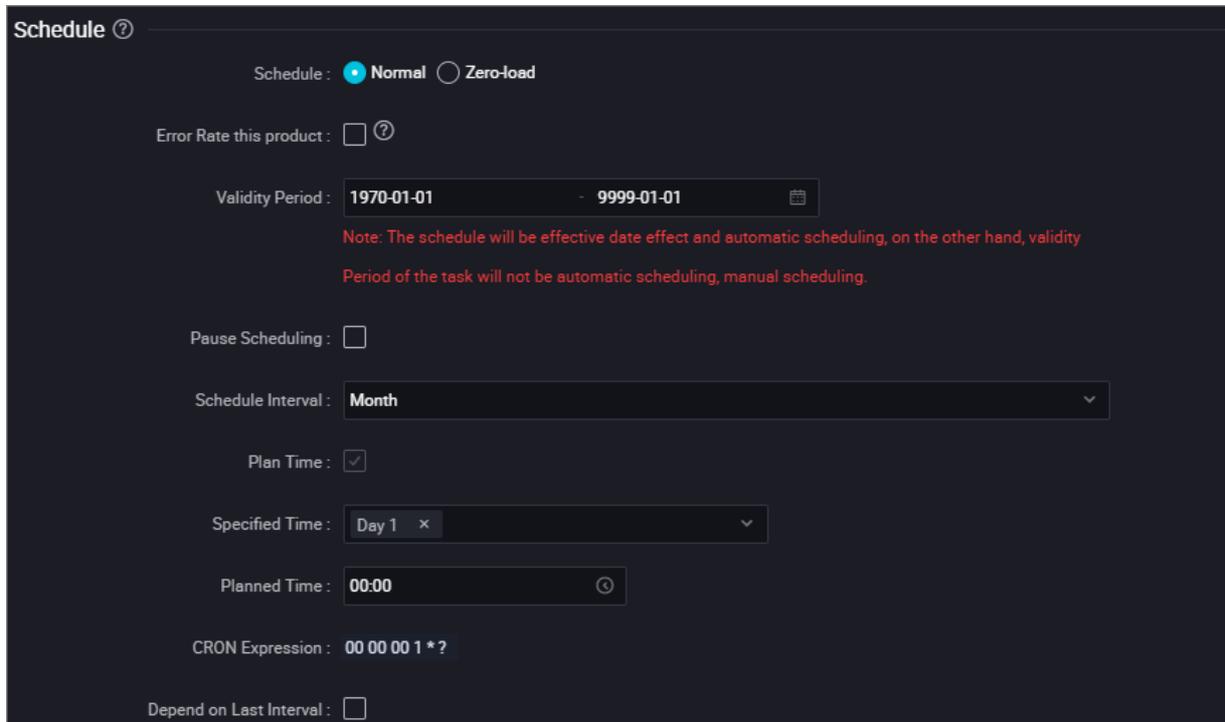
For example, you have created a node. As shown in the preceding figure, the scheduling system runs instances generated on Mondays and Fridays, but returns success responses without running the code for instances generated on Tuesdays, Wednesdays, Thursdays, Saturdays, and Sundays.

Based on the preceding node scheduling properties, the scheduling system automatically generates and runs instances for the node, as shown in the following figure.



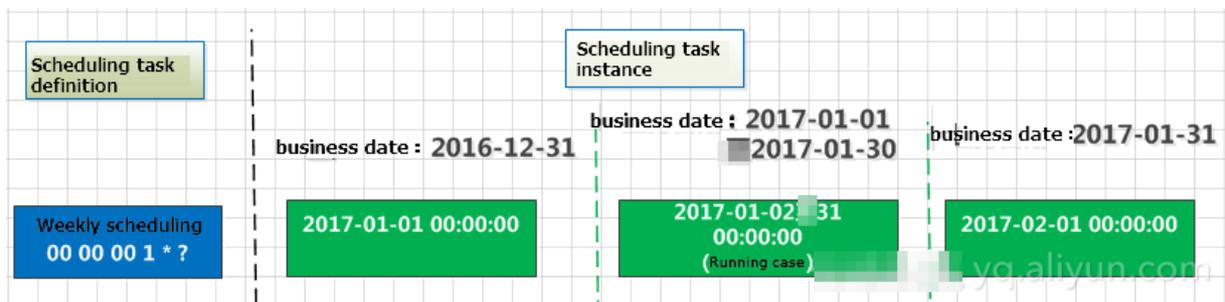
Scheduled by month

Nodes scheduled by month are automatically run at a specified time of specified days every month. On the other days, the scheduling system still generates instances to make sure the proper running of descendant instances. However, the system does not actually run the code or consume resources but directly returns a success response.



For example, you have created a node. As shown in the preceding figure, the scheduling system runs the instance generated on the first day of each month, but returns success responses without running the code for instances generated on the other days.

Based on the preceding node scheduling properties, the scheduling system automatically generates and runs instances for the node, as shown in the following figure.



Scheduled by hour

Nodes scheduled by hour are automatically run once every N hours in a specific time period every day. For example, a node is run once per hour from 01:00 to 04:00 every day.

Note The time period is a closed interval. For example, if a node is scheduled to run once per hour in the period from 00:00 to 03:00, the scheduling system generates four instances every day, which are run at 00:00, 01:00, 02:00, and 03:00, respectively.

Error Rate this product : ?

Validity Period : 1970-01-01 - 9999-01-01

Note: The schedule will be effective date effect and automatic scheduling, on the other hand, validity Period of the task will not be automatic scheduling, manual scheduling.

Pause Scheduling :

Schedule Interval : **Hour**

Plan Time :

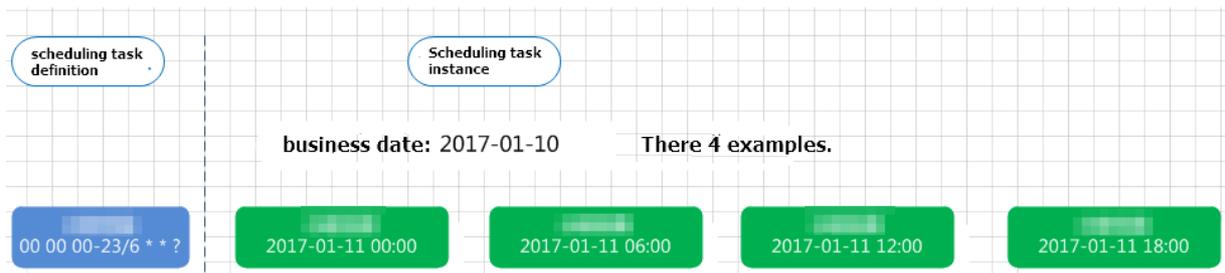
Start Time : 00:00 Interval : 1 h End Time : 23:59

Specified Time : 0:00

CRON Expression : 00 00 00-23/6 ** ?

Depend on Last Interval :

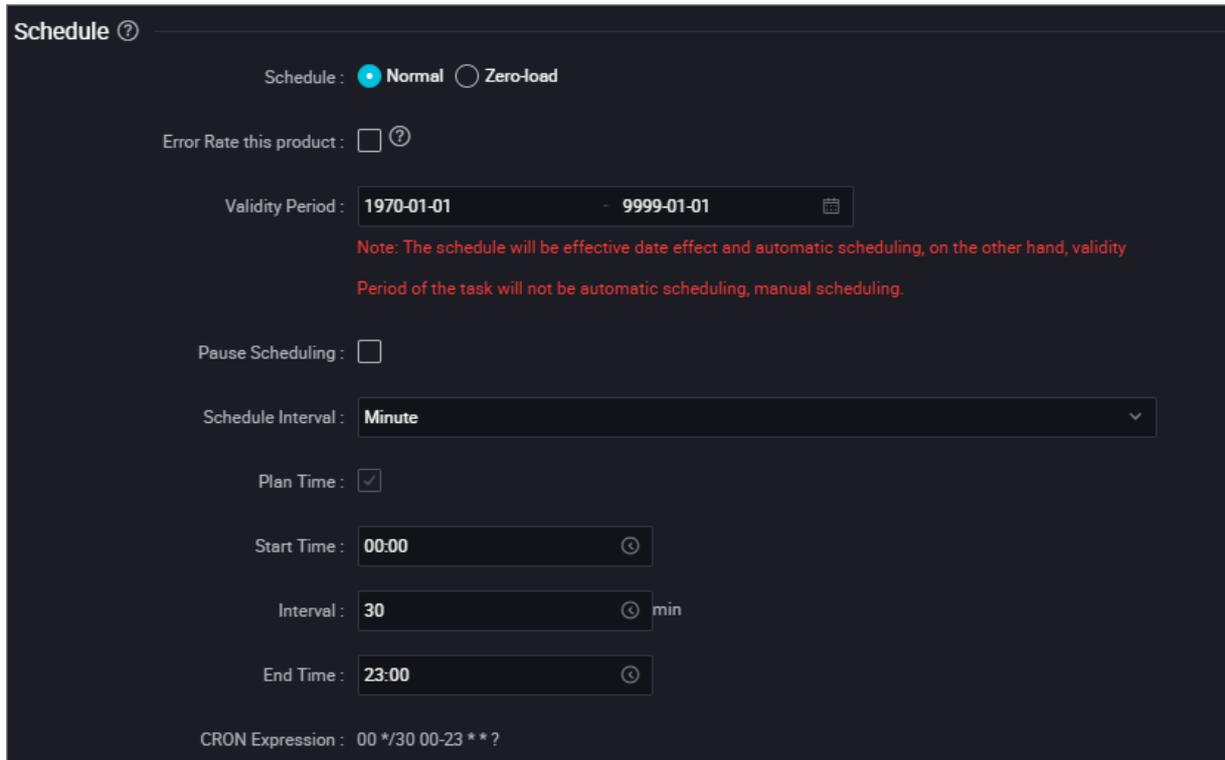
For example, you have created a node. As shown in the preceding figure, the node is automatically run every 6 hours in the period from 00:00 to 23:59 every day. In this case, the scheduling system automatically generates and runs instances for the node, as shown in the following figure.



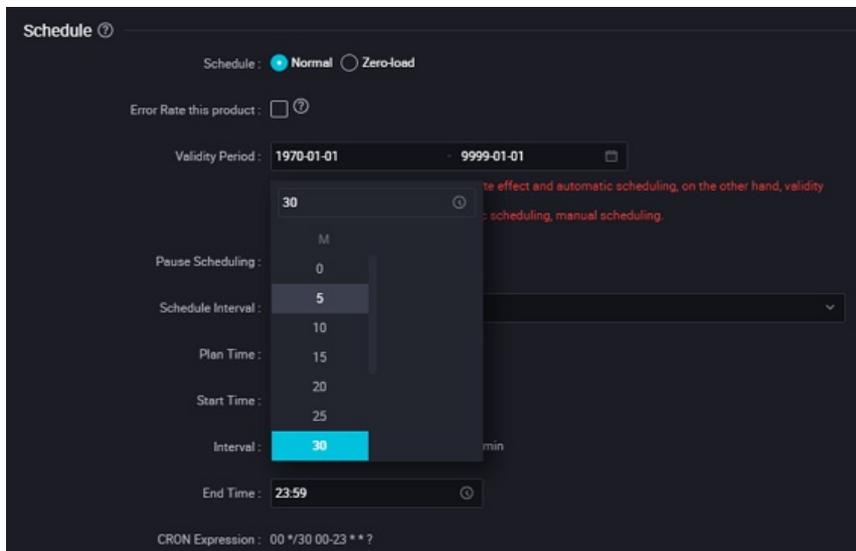
Scheduled by minute

Nodes scheduled by minute are automatically run once every N minutes in a specific time period every day.

For example, you have created a node. As shown in the following figure, the node is run every 30 minutes in the period from 00:00 to 23:00 every day.



Currently, a minimum interval of 5 minutes is supported. The time expression is automatically generated based on the time you select and cannot be modified.



FAQ

- Q: Node A is scheduled by hour and Node B is scheduled by day. How do I enable Node B to automatically run every day after all instances of Node A are run?

A: A node scheduled by day can depend on a node scheduled by hour. To enable Node B to automatically run every day after all 24 instances of Node A are run, do not specify the time to run Node B every day. Then, configure Node A as an ancestor of Node B. For more information, see the Dependencies topic. A node can depend on any other node, regardless of the recurrence. The recurrence of each node is specified in its scheduling properties.

- Q: Node A is run once per hour on the hour every day and Node B is run once per day. How do I enable Node B to automatically run after Node A is run for the first time every day?

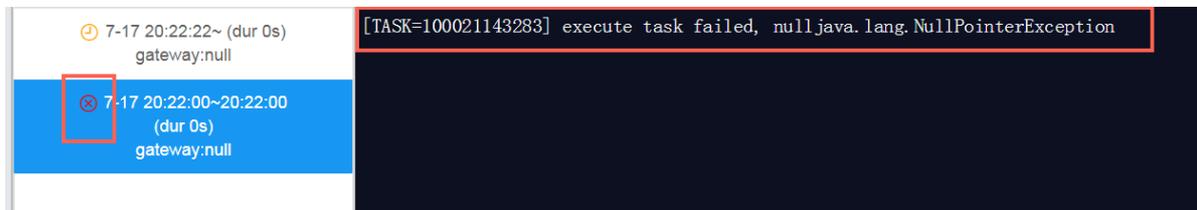
A: When configuring the scheduling properties of Node A, select **Cross-Cycle Dependencies** and select **Instances of Current Node** from the Depend On drop-down list. When configuring the scheduling properties of Node B, configure Node B to depend on Node A and set the scheduled time of Node B to 00:00 every day. In this way, instances of Node B only depend on the instance of Node A generated at 00:00 every day, that is, the first instance of Node A.

- Q: Node A is run once every Monday and Node B depends on Node A. How do I enable Node B to run once every Monday?

A: Set the scheduling properties of Node B to be the same as those of Node A. That is, select **Week** as the instance recurrence and select **Monday**.

- Q: How are the instances of a node affected when the node is deleted?

A: When a node is deleted, its instances are retained because the scheduling system still generates one or more instances for the node based on the scheduling properties. However, when the scheduling system runs such instances after the node is deleted, an error message appears because the required code is unavailable, as shown in the following figure.



- Q: Can I enable a node to process monthly data on the last day of each month?

A: No, DataWorks does not support setting a node to run on the last day of each month. If you enable a node to run on the thirty-first day of each month, the scheduling system runs a node instance in each month that has 31 days and returns a success response without running the code in any other month.

We recommend that you configure a node to process the data of the past month on the first day of each month.

- Q: If a node scheduled by day depends on a node scheduled by hour, how do I enable the node scheduled by day to run at 00:00 every day?

A: You can configure the node scheduled by day to depend on the data generated on the day before for the node scheduled by hour. If the node scheduled by day depends on the data generated on the current day for the node scheduled by hour, the instances of the node scheduled by day can be run only on the next day.

In the Schedule section of the node scheduled by day, select **Cross-Cycle Dependencies**, select **Instances of Custom Nodes** from the Depend On drop-down list, and then enter the ID of the node scheduled by hour on which the node scheduled by day depends. Commit and deploy the node scheduled by day.

- Q: What can I do if I do not know when the output data of the ancestor node is generated?

A: You can set the cross-cycle dependency for the current node to depend on the last-cycle instances of the ancestor node.

- Q: After a modified node is committed and deployed to the production environment, will the node instances that were originally faulty in the production environment be overwritten?

A: No, the node instances that have been generated will not be overwritten. The updated code is used to run the node instances that are newly generated and have not been run. If the scheduling properties are modified, the modified configuration also applies to the newly generated node instances.

2.5.6.4. Dependencies

Scheduling dependencies are the foundation for building orderly workflows. You must configure correct dependencies between nodes to make sure that business data is produced effectively and in a timely manner. This helps standardize data development activities.

DataWorks allows you to automatically parse node dependencies from the code or manually customize node dependencies. You can configure correct relationships between ancestor and descendant nodes and monitor the running status of nodes to make sure the orderly production of business data.

The purpose of configuring node dependencies is to check the data output time of the table queried by SQL statements and check whether data is properly produced from an ancestor node based on the node status.

You can set the output of an ancestor node as the input of a descendant node to configure a dependency between the two nodes.

Regardless of the dependency configuration mode, the overall scheduling logic is that descendant nodes can be run only after ancestor nodes are run. Therefore, each node in a workflow must have at least one parent node. The dependencies between the parent nodes and child nodes are the core of scheduling dependencies. The following sections describe the principles and configuration methods of scheduling dependencies in detail.

Differences between automatic parsing and custom dependencies

DataWorks can automatically parse the input and output of a node based on the lineage parsed from the code.

If the lineage parsed from the code is inaccurate, you can add custom dependencies as needed. We recommend that you write the code correctly to parse the lineage from the code and reduce custom dependencies. The following example shows how to configure the input and output of a node.

Auto Parse: If you select Yes, node dependencies are automatically parsed from the code.

For example, the code of an ODPS SQL node is as follows:

```
insert overwrite table table_a as select * from project_b_name.table_b;
```

From the code, DataWorks determines that the current node depends on the node that generates table_b and the current node generates table_a. Therefore, the output name of the parent node is project_b_name.table_b and the output name of the current node is project_name.table_a.

- If you do not want to parse node dependencies from the code, select **No** for Auto Parse.
- If a table in an SQL statement is both an output table and a referenced table on which another node depends, the table is parsed only as an output table.
- If a table in an SQL statement is used as an output table or a referenced table multiple times, only one scheduling dependency is parsed.
- If the SQL code contains a temporary table, the table is not involved in a scheduling dependency. Temporary tables are prefixed with t_. For more information, see [Project Configuration](#).

Parent nodes

In the Dependencies section of a node, you must specify an ancestor node as the parent node on which the current node depends. You must enter the output name of the ancestor node, rather than the ancestor node name. A node may have multiple output names. Enter an output name as needed. You can search for an output name of the ancestor node to be added, or click Parse I/O to parse the output name based on the lineage parsed from the code.

 **Note** You must enter an output name or output table name to search for the ancestor node.

If you enter an output name to search for the ancestor node, DataWorks searches for the output name among the output names of nodes that have been committed to the scheduling system.

- Search by entering an output name

You can enter an output name to search for the ancestor node and configure the node as the parent node of the current node to create a dependency.

- Search by entering an output table name

When using this method, make sure that the entered output table name of the ancestor node is the table name used in the INSERT or CREATE statement of the current node, such as Project name.Table name . Such output names can be automatically parsed.

After you click the **Submit** icon, the output table name of the parent node configured for the current node can be found when you enter an output table name to search for the ancestor node for other nodes.

Outputs

You can click the **Properties** tab in the right-side navigation pane to view and configure the output of the current node.

DataWorks assigns a default output name that ends with .out to each node. You can also customize an output name or click Parse I/O to parse the output name based on the lineage parsed from the code.

 **Note** The output name of each node must be globally unique.

FAQ

- Q: After DataWorks automatically parses the input and output of a node, the node fails to be committed. An error message appears to indicate that the parsed output name workshop_yanshi.tb_2 of the parent node does not exist and you must commit the parent node before committing the current node. Why does this error occur?

A: The possible causes are as follows:

- The ancestor node is not committed. Commit the ancestor node and try again.
- The ancestor node is committed, but workshop_yanshi.tb_2 is not an output name of the ancestor node.

 **Note** Usually, the output names of the parent node and the current node are automatically parsed based on the table name that is used in the INSERT or CREATE statement or follows the FROM keyword. Make sure that you follow the principles of automatic parsing in the Differences between automatic parsing and custom dependencies section.

- Q: In the output of the current node, the descendant node name and ID are empty and cannot be specified. Why does this happen?

A: If the current node does not have a descendant node, the descendant node name and ID are empty. After a descendant node is configured for the current node, the corresponding content can be automatically parsed.

- Q: What is the output name of a node used for?

A: The output name of a node is used to establish dependencies between nodes. For example, if the output name of Node A is ABC and Node B uses ABC as its input, a dependency is established between nodes A and B.

- Q: Can a node have multiple output names?

A: Yes, a node can have multiple output names. If a descendant node references an output name of the current node as the output name of the parent node, a dependency is established between the descendant node and the current node.

- Q: Can multiple nodes have the same output name?

A: No, the output name of each node must be unique under your Apsara Stack tenant account. If multiple nodes export data to the same MaxCompute table, we recommend that you use Table name_Partition ID as the output name format of these nodes.

- Q: How can I avoid intermediate tables when I enable DataWorks to automatically parse node dependencies?

A: Right-click an intermediate table name in the SQL code and select **Delete Input** or **Delete Output**. Then, click Parse I/O to parse the input and output of the node.

- Q: How do I configure dependencies for the upmost node in a workflow?

A: You can set the node to depend on the root node of the current workspace.

- Q: Why do I find a non-existent output name of Node B when I enter an output name to search for the ancestor node for Node A?

A: DataWorks searches for the output name among the output names of nodes that have been committed to the scheduling system. After Node B is committed, if you delete the output name of Node B and does not commit Node B to the scheduling system again, the deleted output name of Node B can still be found.

- Q: How do I enable nodes A, B, and C to run in sequence once per hour?

A: Set the output of Node A as the input of Node B and the output of Node B as the input of Node C. Also, set nodes A, B, and C to run once per hour.

- Q: An error message is returned to indicate that the parent node ID fails to be automatically parsed based on an output table name. Why does this error occur?

A: This error does not indicate that the table does not exist. Instead, it indicates that the table is not the output of a specific node. Therefore, the table name cannot be used to find the node that generates the table data. In this case, the dependency on the node cannot be created.

According to the principles of automatic parsing described in this topic, a dependency is created after the output of an ancestor node is set as the input of a descendant node. If no ancestor node can be parsed based on the xc_demo_partition table referenced in SQL statements, no node uses the xc_demo_partition table as its output.

You can resolve this problem in the following way:

- i. Find the node that generates the table data and view the node output.

If you do not know which the target node is, you can enter keywords to search the code for the node in fuzzy match mode.

- ii. If the table data is uploaded from a local server or you do not need to depend on the node, you can right-click the table name in the code and select **Delete Input**.

 **Note** We recommend that you write the code correctly to parse the lineage from the code and reduce custom dependencies.

2.5.7. Components

2.5.7.1. Create a script template

This topic describes the definition and composition of script templates and how to create a script template.

Definition

A script template defines an SQL code process that involves multiple input and output parameters. Each SQL code process references one or more source tables. You can filter source table data, join source tables, and aggregate them to generate a result table required for new business.

Value

In actual business, many SQL code processes are similar. The input and output tables in these processes may have the same or compatible schema but different names. In this case, developers can abstract an SQL code process as a script template to reuse the SQL code. The script template extracts input parameters from input tables and generates output parameters in output tables.

To create SQL script templates, you can select script templates from the script template list based on your business process and configure specific input and output tables in your business for the selected script templates, without repeatedly copying the code. This greatly improves the development efficiency and avoids repeated development. You can deploy and run the created SQL script templates in the same way as other SQL nodes.

Composition

Similar to a function, a script template consists of input parameters, output parameters, and an SQL code process.

Input parameters

The input parameters of a script template have the properties such as the parameter name, parameter type, parameter description, and parameter definition. The parameter type can be table or string.

- A table-type parameter specifies the table to be referenced in an SQL code process. When you use a script template, you can specify the input table required for the specific business.
- A string-type parameter specifies the variable control parameter in an SQL code process. For example, to export only the sales amount of the top N cities in each region in a result table of an SQL code process, you can use a string-type parameter to specify the value of N.

To export the total sales amount of a province in a result table of an SQL code process, you can set a string-type parameter to specify the province and obtain the sales data of the specified province.

- The parameter description specifies the role of a parameter in an SQL code process.
- The parameter definition is a text definition of the table schema, which is required only for table-type parameters. When you specify the parameter definition for a table-type parameter, you must provide an input table that contains the same field names and compatible types defined by the table-type parameter so that the SQL code process can run properly. Otherwise, an error is returned when the SQL code process runs because the specified field name cannot be found in the input table. The input table must contain the field names and types defined by the table-type parameter. The input table can also contain other fields. The field names and types in the input table can be in any order. The parameter definition is for reference only.
- We recommend that you enter the parameter definition in the following format:

```
Name of field 1 Type of field 1 Description of field 1  
Name of field 2 Type of field 2 Description of field 2  
Name of field n Type of field n Description of field n
```

Examples:

```
area_id string 'Region ID'  
city_id string 'City ID'  
order_amt double 'Order amount'
```

Output parameters

- The output parameters of a script template have the properties such as the parameter name, parameter type, parameter description, and parameter definition. The parameter type must be table. A string-type output parameter has no logical meaning.
- A table-type parameter specifies the table to be generated in an SQL code process. When you use a script template, you can specify the result table that the SQL code process generates for the specific business.
- The parameter description specifies the role of a parameter in an SQL code process.
- The parameter definition is a text definition of the table schema. When you specify the parameter definition for a table-type parameter, you must provide an output table that contains the same number of fields and compatible types defined by the table-type parameter so that the SQL code process can run properly. Otherwise, an error is returned when the SQL code process runs because the number of fields does not match or the field type is incompatible. The field names of the output table do not need to be consistent with those defined by the table-type parameter. The parameter definition is for reference only.
- We recommend that you enter the parameter definition in the following format:

```
Name of field 1 Type of field 1 Description of field 1  
Name of field 2 Type of field 2 Description of field 2  
Name of field n Type of field n Description of field n
```

Examples:

```
area_id string 'Region ID'  
city_id string 'City ID'  
order_amt double 'Order amount'  
rank bigint 'Ranking'
```

SQL code process

The parameters in an SQL code process are referenced in the following format: `@@{Parameter name}`.

By containing an abstract SQL code process, a script template controls and processes an input table based on input parameters to generate an output table with business value.

To develop an SQL code process, you must use input and output parameters in the code properly to make sure that they can be set as needed and correct SQL code can be generated and run during the process.

Create a script template

1. [Log on to the DataWorks console](#).
2. On the left-side navigation submenu, click the **Snippets** icon.
3. On the Snippets tab, move the pointer over  and choose **Create > Snippet**.
4. In the **Create Snippet** dialog box, set **Snippet Name**, **Description**, and **Location**.
5. Click **Commit**.

Source table schema

The following table describes the schema of a source MySQL table that contains sales data.

Field	Data type	Description
order_id	varchar	The ID of the order.
report_date	datetime	The date of the order.
customer_name	varchar	The name of the customer.
order_level	varchar	The level of the order.
order_number	double	The number of orders.
order_amt	double	The amount of the order.
back_point	double	The discount.
shipping_type	varchar	The transportation method.
profit_amt	double	The amount of the profit.
price	double	The unit price.
shipping_cost	double	The transportation cost.
area	varchar	The region.
province	varchar	The province.
city	varchar	The city.
product_type	varchar	The type of the product.
product_sub_type	varchar	The subtype of the product.
product_name	varchar	The name of the product.
product_box	varchar	The packaging of the product.
shipping_date	datetime	The shipping date.

Business implication

Script template name: get_top_n

This script template uses the specified sales data table as the table-type input parameter, the number of the top cities as the string-type input parameter, and the total sales amount of the cities for ranking. By using this SQL code process, you can obtain the rankings of the specified top cities in each region with ease.

Script template parameters

Input parameter 1

- Parameter name: myinputtable
- Type: table

Input parameter 2

- Parameter name: topn
- Type: string

Output parameter 3

- Parameter name: myoutput
- Type: table

Parameter definition:

- area_id string
- city_id string
- order_amt double
- rank bigint

You can execute the following statement to create a table for storing the sales data of a specified number of top cities:

```
CREATE TABLE IF NOT EXISTS company_sales_top_n
(
  area STRING COMMENT 'Region',
  city STRING COMMENT 'City',
  sales_amount DOUBLE COMMENT 'Sales amount',
  rank BIGINT COMMENT 'Ranking'
)
COMMENT 'Company sales rankings'
PARTITIONED BY (pt STRING COMMENT '')
LIFECYCLE 365;
```

Example of defining an SQL code process

```

INSERT OVERWRITE TABLE @@{myoutput} PARTITION (pt='${bizdate}')
  SELECT r3.area_id,
  r3.city_id,
  r3.order_amt,
  r3.rank
from (
SELECT
  area_id,
  city_id,
  rank,
  order_amt_1505468133993_sum as order_amt ,
  order_number_15054681****_sum,
  profit_amt_15054681****_sum
FROM
  (SELECT
  area_id,
  city_id,
  ROW_NUMBER() OVER (PARTITION BY r1.area_id ORDER BY r1.order_amt_1505468133993_sum DESC)
AS rank,
  order_amt_15054681****_sum,
  order_number_15054681****sum,
  profit_amt_1505468****_sum
FROM
  (SELECT area AS area_id,
  city AS city_id,
  SUM(order_amt) AS order_amt_1505468****_sum,
  SUM(order_number) AS order_number_15054681****_sum,
  SUM(profit_amt) AS profit_amt_1505468****_sum
FROM
  @@{myinputtable}
WHERE
  SUBSTR(pt, 1, 8) IN ( '${bizdate}' )
GROUP BY
  area,
  city )
r1 ) r2
WHERE
  r2.rank >= 1 AND r2.rank <= @@{topn}
ORDER BY
  area_id,
  rank limit 10000) r3;

```

Sharing scope

Script templates can be shared within a workspace or made public.

By default, a deployed script template is visible and available to users within the current workspace. The developer of a script template can click the **Publish Snippet** icon to make the general-purpose script template public to the current tenant account so that all users under the account can view and use the script template.

You can view the **Publish Snippet** icon in the toolbar of the configuration tab of a script template. If the icon is clickable, the script template is made public.

Use of script templates

For more information about how to use a developed script template, see [Use components](#).

Reference records

In the script template list, double-click a script template. On the configuration tab that appears, click the **Snippet Nodes** tab in the right-side navigation pane to view the reference records of the script template.

2.5.7.2. Use a script template

To improve development efficiency, you can create data analytics nodes by using the script templates provided by workspace members and tenants.

Note the following points when you use script templates:

- The script templates provided by members of the current workspace are available on the **Workspace-Specific** tab.
- The script templates provided by tenants are available on the **Public** tab.

GUI elements

Icon or tab	Description
Save icon	Saves the settings of the current script template.
Steal Lock icon	Allows you to steal the lock of the current script template and then edit it if you are not the owner of the script template.
Submit icon	Commits the current script template to the development environment.
Publish Snippet icon	Makes the current general-purpose script template public to the current tenant account so that all users under the account can view and use the script template.
Parse I/O Parameters icon	Parses input and output parameters from the code. <div style="background-color: #e0f2f7; padding: 5px;"><p> Note Typically, the parameters entered here are table names instead of scheduling parameters.</p></div>
Run icon	Runs the current script template in the development environment.
Stop icon	Stops running the current script template.
Format icon	Formats the code based on keywords.
Parameters tab	Allows you to view the basic information and set input and output parameters for the current script template.
Versions tab	Allows you to view the deployed versions of the current script template.
Snippet Nodes tab	Lists the reference records of the current script template.

2.5.8. Custom node type

2.5.8.1. Overview

DataStudio supports default node types such as ODPS SQL and Shell nodes. You can also create custom node types to meet your requirements.

To create a custom node type, you need to create a custom wrapper and use it to define a custom node type.

Entry

1. Log on to the DataWorks console.
2. Click **Node Market** in the upper-right corner to go to the node configuration page.

 **Note** Only the workspace owner and administrators can access this page.

View the list of wrappers

The Wrappers page displays all the wrappers you have created. You can click **Create** in the upper-right corner to create a custom wrapper.

The values displayed in the Latest Version, Version in Development, and Version in Production Environment columns for the created wrappers follow these rules:

- If a created wrapper has not been deployed, the values of both the Version in Development and Version in Production Environment columns are **Not Deployed**.
- If a wrapper has been deployed, the version and the deployment time appear in these columns.
- If a wrapper is under deployment, the values of both the Version in Development and Version in Production Environment columns are **Deploying**.

You can click **Settings**, **View Versions**, or **Delete** in the Actions column of each wrapper.

Action	Description
Settings	You can click Settings to configure the wrapper. The page that appears depends on the wrapper status. The Deploy in Production Environment page appears if the wrapper has been deployed in the production environment.
View Versions	<p>You can click View Versions to view all historical versions of the wrapper.</p> <ul style="list-style-type: none"> • View: You can click this button to view the settings of the selected version. • Roll Back: You can click this button to roll back to the selected version. After you click this button, the system creates a new version for the wrapper. In the new version, the wrapper uses the basic settings and the resource file of the selected version. The new version number equals the latest version number among all the versions plus 1. • Download: You can click Download to download the resource file of the selected version.
Delete	<p>If an error occurs while a node type is using the wrapper, you need to delete the node type.</p> <p> Note Before deleting a wrapper, ensure that no node type is associated with the wrapper.</p>

Create a custom wrapper

A wrapper is the core processing logic of a node type. For example, after you compile an SQL statement in the editor for an ODPS SQL node and submit the statement, the system calls the corresponding wrapper to parse and run the statement. You need to create a wrapper before creating a custom node type. Currently, only the Java programming language is supported.

The procedure of creating a wrapper includes four steps: specify settings for the wrapper, deploy the wrapper in the development environment, test the wrapper in the development environment, and deploy the wrapper in the production environment. For more information, see [Create a custom wrapper](#).

View the list of custom node types

The Custom Node Types page displays all custom node types in the workspace. You can click **Create** in the upper-right corner to create a custom node type. For more information, see [Create a custom node type](#).

Currently, you can only create custom node types in DataStudio.

The workspace owner or node type creator can **change** or **delete** existing node types.

- **Change:** You can click **Change** to edit the settings for the node type as needed.
- **Delete:** You can click this button to delete the node type that no node uses. If any node uses the node type, a message appears, indicating that you need to disable the node first before deleting the node type.

Use a custom node type

After creating a custom node type, go to the **Data Analytics** page.

Move the pointer over the **Create** icon and click **Data Analytics**. In the list that appears, select the created node type to create a node.

2.5.8.2. Create a custom wrapper

The procedure of creating a wrapper includes four steps: specify settings for a wrapper, deploy the wrapper in the development environment, test the wrapper in the development environment, and deploy the wrapper in the production environment.

Specify settings for a wrapper

1. Click **Wrappers** in the left-side navigation pane. On the page that appears, click **Create** in the upper-right corner.
2. Specify the parameters in the **Settings** step.

Parameter	Description
Name	The name of the wrapper. It must start with a letter and can only contain letters, digits, and underscores (_).
Owner	The owner of the wrapper. You can select an owner from the workspace members. You are not allowed to edit wrappers owned by other members even if you are an administrator. Only the workspace owner can edit the wrappers of other members.
Resource Type	The type of the resource package for configuring the wrapper. Valid values: JAR and Archive . The size of the resource package can be up to 50 MB.

Parameter	Description
Resource File	The local resource file or OSS object for configuring the wrapper. <div style="background-color: #e1f5fe; padding: 5px;"> <p> Note The size of a local file can be up to 50 MB, and the size of a file that is stored in an OSS bucket can be up to 200 MB.</p> </div>
Class Name	The full path of the class for implementing the user wrapper.
Parameter Example	The parameters designed based on the JAR package you upload.
Version	The version of the configured wrapper. Select Create Version if you are creating a new wrapper. Select Overwrite Version if you are editing and rolling back a version. <div style="background-color: #e1f5fe; padding: 5px;"> <p> Note The version number is automatically generated.</p> </div>
Description	The description of the wrapper version.

3. Click **Save** and then click **Next**.

-  **Note** The settings are updated to the database after you click **Save**.

 - If you only modify basic settings of a wrapper without changing the resource file, the modification takes immediate effect after you click **Save**.
 - If you change the resource file, the change only applies after deployment.

Deploy the wrapper in the development environment

After you specify the parameters in the **Settings** step and click **Next**, the information in the **Deploy in Development Environment** step is updated accordingly. You can identify the changes by checking the file name and MD5 checksum.

Click **Deploy in Development Environment**. You can view the **deployment progress** in real time. After the wrapper is deployed, click **Next**.

Test the wrapper in the development environment

Specify the parameters for testing and click **Test** to send the parameters to the wrapper. This step is to validate the deployment and logic of the wrapper. You can also locally test the wrapper before uploading it for deployment.

After the test, review the output logs in the **Test Results** section on the right to determine whether the test is passed. If the test is passed, select **Test Passed** and click **Next**.

-  **Note** The **Next** button is operable only after you select **Test Passed**.

Deploy the wrapper in the production environment

Click **Deploy in Production Environment**. In the **Confirm** dialog box that appears, click **OK**. The wrapper is deployed in the production environment. You can view the deployment progress in real time.

 **Note** The wrapper to be deployed in the production environment must be of the latest version, have been deployed in the development environment, and have passed the test. Otherwise, a message appears, indicating that the deployment in the production environment fails.

Click **Complete**. You can view and edit the created wrapper on the **Wrappers** page.

2.5.8.3. Create a custom node type

The Configure Custom Node Type page consists of three sections: Basic Information, Interaction, and Wrapper.

1. On the DataStudio page, click **Node Market** in the top navigation bar. On the page that appears, click **Custom Node Types** in the left-side navigation pane.
2. Click **Create** in the upper-right corner.
3. Specify the parameters in the **Basic Information** section.

Parameter	Description
Name	The name of the node type, which cannot be changed after being saved. Each node type has a unique name within the workspace. The name can be up to 20 characters in length, and can only contain letters, spaces, and underscores (_).
Icon	The icon of the node type.
Tabs	The template of the node type. Currently, only Data Analytics is available.
Folder	The folder where the node type belongs. You can select Data Integration or Data Analytics .

4. Specify the parameters in the **Interaction** section.

Parameter	Description
Shortcut Menu	<ul style="list-style-type: none"> ◦ The options to appear in the shortcut menu. The following options are selected by default: Rename, Move, Clone, Steal Lock, View Versions, Locate in Operation Center, Delete, and Submit for Review. ◦ More options include Edit, Copy Resource Name, and Send to DataWorks Desktop (Shortcut).
Tool Bar	<ul style="list-style-type: none"> ◦ The options to appear in the top navigation bar. The following options are selected by default: Save, Commit, Commit and Unlock, Steal Lock, Run, Show/Hide, Run with Arguments, Stop, Reload, Run Smoke Test in Development Environment, View Smoke Test Log in Development Environment, Run Smoke Test, View Smoke Test Log, Go to Operation Center of Development Environment, and Format. ◦ More options include Operation Center, Deploy, and Precompile.
Editor Type	The type of the editor. Currently, only Editor Only is available.
Right-Side Bar	<ul style="list-style-type: none"> ◦ The options to appear in the right-side bar. The following options are selected by default: Code Structure and Properties. ◦ More options include Version, Lineage, and Parameters.

Parameter	Description
Auto Parse Option	Specifies whether to display the Auto Parse option for this type of node. If you turn on this switch, the Auto Parse option is displayed on the Properties tab. Otherwise, it is not displayed. In an automatic parsing process, the system parses the input and output of a node based on the lineage specified in the code.

5. Specify the parameters in the **Wrapper** section.

Parameter	Description
Wrapper	The wrapper used for running the type of node. Select a wrapper that has been deployed.
Editor Language	The language used for writing the code in the editor. Currently, only ODPS SQL is available.
Use MaxCompute as Engine	Specifies whether to use MaxCompute as the compute engine. If your wrapper uses MaxCompute as the compute engine, select Yes. Otherwise, select No. Default value: Yes.

6. Click **Save and Exit**. Then, go to the **Data Analytics** page to use the custom node type that is created.

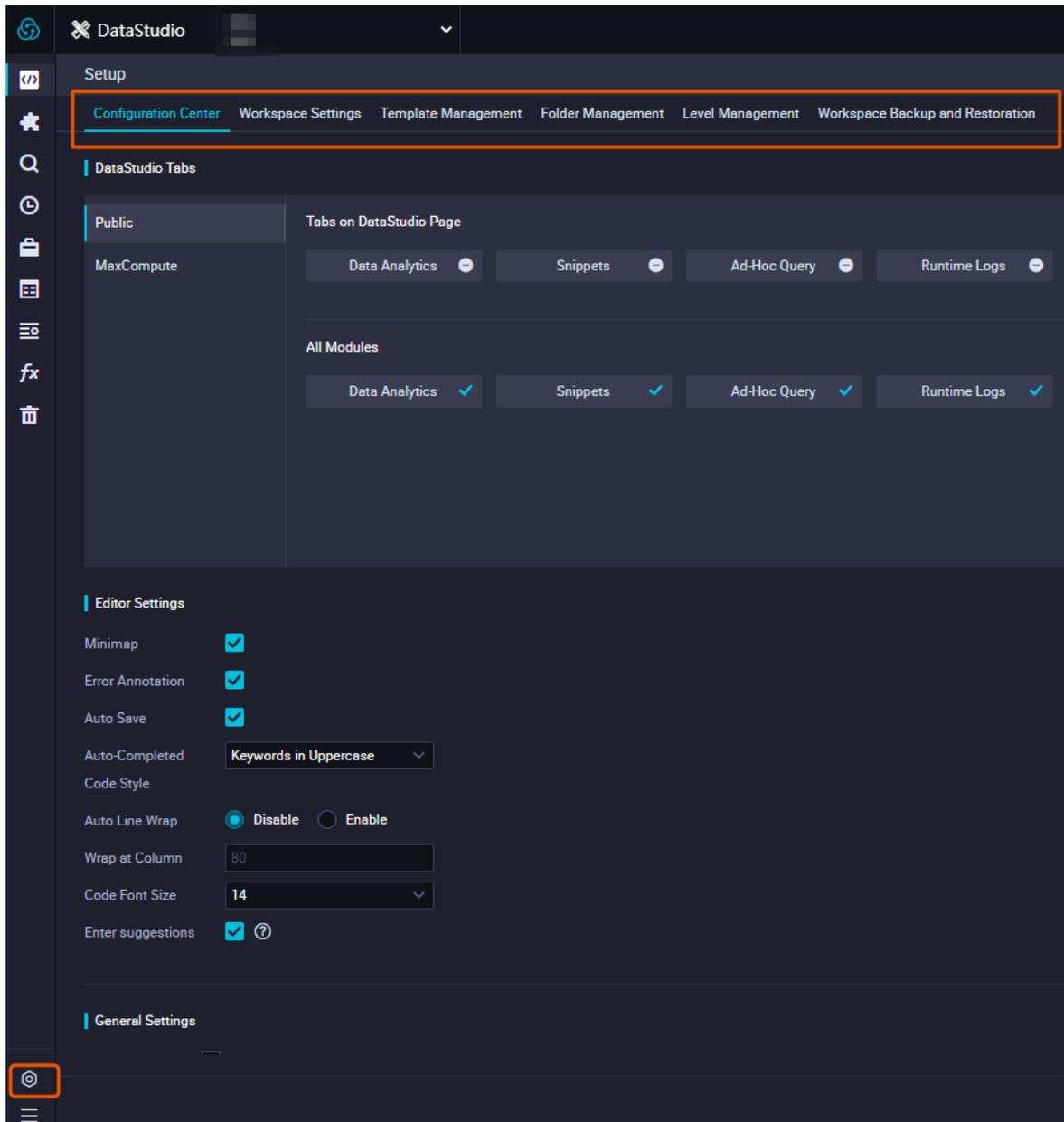
2.5.9. Manage configurations

2.5.9.1. Setup

On the Setup page, you can add and delete modules. You can also configure code templates, folders, and table levels on this page.

Procedure

1. [Log on to the DataWorks console](#).
2. Click  in the lower-left corner of the **Data Studio** page to go to the **Setup** page.



You can perform operations on the following tabs:

- [Configuration Center](#)
- [Project Configuration](#)
- [Template management](#)
- [Folder management](#)
- [Level management](#)

2.5.9.2. Configuration center

You can combine your DataStudio modules and specify editor settings on the Configuration Center tab.

Go to Configuration Center

1. [Log on to the DataWorks console.](#)

- Click  in the lower-left corner of the **DataStudio** page. The **Configuration Center** tab appears.

The Configuration Center tab includes three sections: **DataStudio Tabs**, **Editor Settings**, and **General Settings**. After the configurations are completed, you can click **Apply to All Workspaces** in the lower-right corner of the page to apply the settings to all existing workspaces.

DataStudio tabs

On the **DataStudio Tabs** tab, you can add and delete public and MaxCompute functional modules, and drag modules to change their orders.

- Under **Tabs on DataStudio Page**, click  next to a module to delete it. Deleted modules will not appear in the left-side navigation pane of the **DataStudio** page.
- Under **All Modules**, click the desired module to add it. Added modules will appear in the left-side navigation pane of the **DataStudio** page.

 **Note** The module settings take effect immediately for the current workspace. To make the module settings effective for all workspaces, click **Apply to All Workspaces** in the lower-right corner of the page.

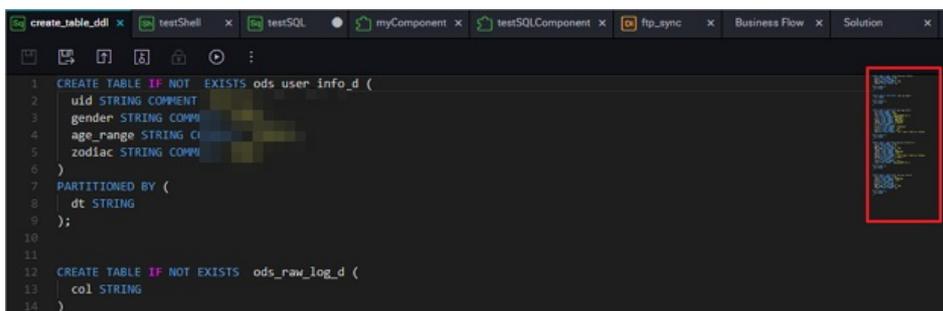
Editor settings

You can configure the code editor in the Editor Settings section. The editor settings take effect immediately for the current workspace without requiring you to refresh the page.



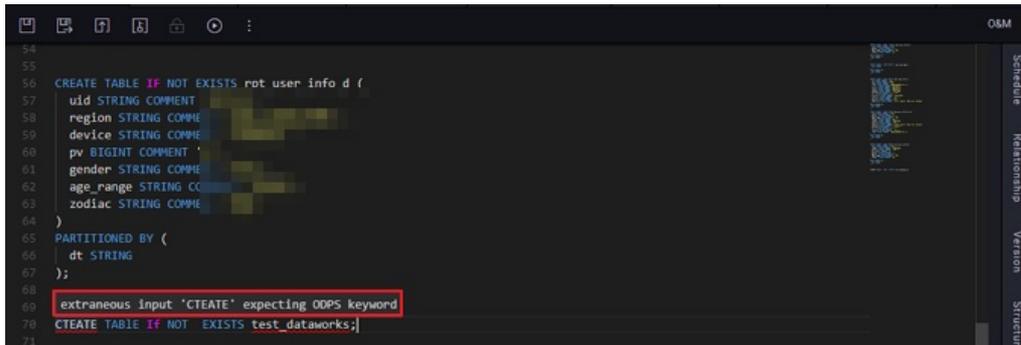
- Minimap**

The masked code in the current interface is displayed in the minimap in the upper-right corner of the page. When the code is long, you can move the pointer to specify the code block to be displayed in the minimap.



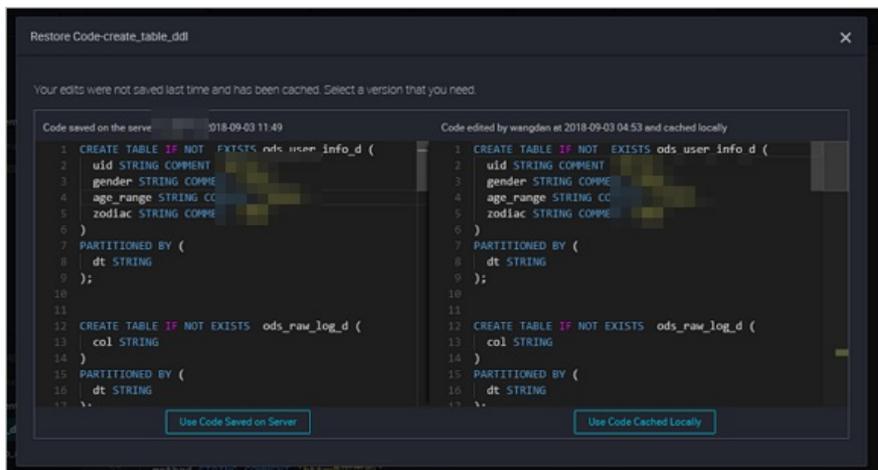
- Error Annotation**

If you select this check box, DataWorks marks potential syntax errors with a red squiggly line. When you see a syntax error, you can move the pointer over the underlined code to view the error message.



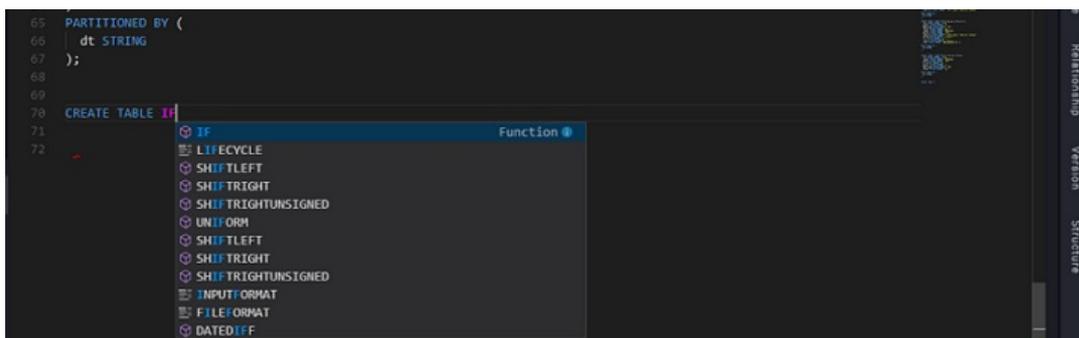
- **Auto Save**

If you select this check box, DataWorks automatically saves the code being edited at a specific interval. In this way, if the code editor of a node is closed unexpectedly, you can click **Use Version Saved on Server** or **Use Version Saved in Local Cache** to re-open the node.



- **Auto-Completed Code Style**

You can set the code style to uppercase or lowercase as required.



- **Auto Line Wrap**

You can set Auto Line Wrap to **Disable** or **Enable**.

- **Wrap at Column**

- If Auto Line Wrap is set to **Disable**, the value of Wrap at Column is 80 by default, which cannot be modified.
- If Auto Line Wrap is set to **Enable**, you can set a value for Wrap at Column as required.

- **Code Font Size**

Valid values: 12 to 18. You can change the font size based on your habits and code size.

- **Enter suggestions**

If you select this check box, the system automatically displays suggestions on how to set a field when you press Enter. If you clear this check box, a new line is started after you press Enter. In addition to the Enter key, you can also use the Tab key to enter prompted suggestions.

- **Auto Completion**

You can specify whether to enable the following code hints when you enter the code:

- **Continuous Smart Tips**: specifies whether to automatically add a space after each auto-completed term such as a keyword, table name, or field name.
- **Keyword**: specifies whether to enable keyword hints.
- **Syntax Template**: specifies whether to enable syntax template hints.
- **Project**: specifies whether to enable project name hints.
- **Table Name**: specifies whether to enable table name hints. When this feature is enabled, the system gives higher priority to tables used recently.
- **Field**: specifies whether to enable field name hints.

General settings

- **Display Node Engine Information on DAG**

You can specify whether to display node engine information on the DAG.

- **Theme**

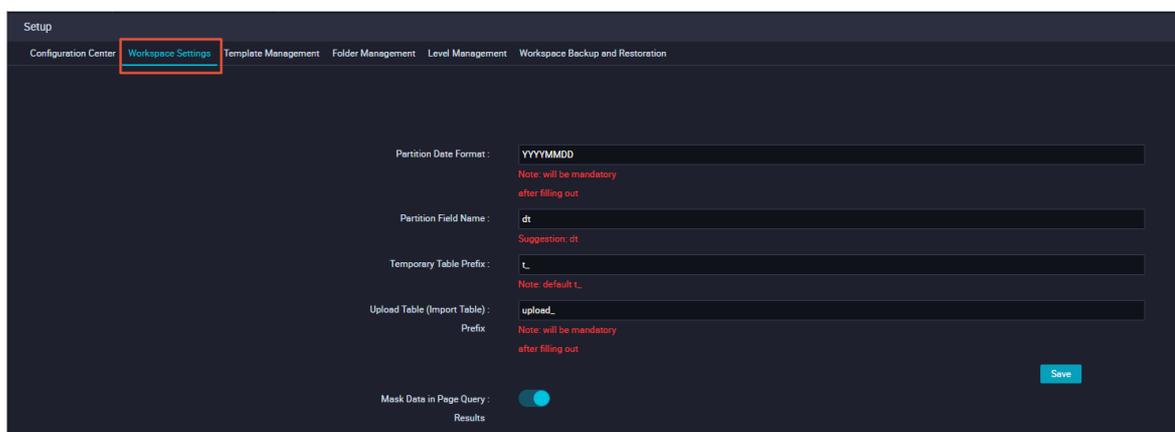
You can set the DataStudio theme to black or white.

2.5.9.3. Workspace settings

The Workspace Settings tab displays five parameters: Partition Date Format, Partition Field Name, Temporary Table Prefix, Upload Table (Import Table) Prefix, and Mask Data in Page Query Results.

Go to the Workspace Settings tab

1. [Log on to the DataWorks console.](#)
2. On the **Data Analytics** tab, click  in the lower-left corner.
3. On the **Setup** page, click the **Workspace Settings** tab.



Parameter	Description
Partition Date Format	The default date format of partition field values. You can modify the format as required.
Partition Field Name	The default name of a partition field.
Temporary Table Prefix	The prefix of temporary table names. By default, tables with the prefix t_ in their names are identified as temporary tables.
Upload Table (Import Table) Prefix	The prefix of the names of tables uploaded on the DataStudio page.
Mask Data in Page Query Results	Specifies whether to de-identify data in the query results. When the switch is turned on, the result returned for an ad hoc query node in the current workspace will be de-identified.

Enable de-identification for DataWorks workspaces

Data de-identification for DataWorks needs to be enabled in workspaces one by one. After data de-identification is enabled, the result returned for an ad hoc query node in the current workspace will be de-identified. The underlying storage data is not affected because only dynamic de-identification is performed.

Note For example, data de-identification is enabled in workspace A but not workspace B. If you initiate a request in workspace B to query tables in workspace A, the query result is displayed in plaintext.

On the **Workspace Settings** tab, turn on **Mask Data in Page Query Results**. After you click **Save**, the result returned for an ad hoc query node in the current workspace will be de-identified.

Note By default, the **Mask Data in Page Query Results** switch is turned off and you are not allowed to download de-identified data.

After data de-identification is enabled for DataWorks workspaces, the data types listed in the following table are de-identified by default.

Type	Data de-identification rule	Raw data	De-identified data
ID card number	Only the first and last digits in a 15-digit or an 18-digit ID card number are displayed in plaintext. All the other digits are displayed as asterisks (*).	512345678943215678	5*****8
Mobile number	Only the first three and last two digits in a mobile number in mainland China are displayed in plaintext. All the other digits are displayed as asterisks (*).	18112345678	181*****78

Type	Data de-identification rule	Raw data	De-identified data
Email address	If the string before the at sign (@) in an email address contains three or more characters, only the leftmost three characters are displayed in plaintext, followed by three asterisks (*). If the string before the at sign (@) contains only one or two characters, the entire string is displayed in plaintext, followed by three asterisks (*).	<ul style="list-style-type: none"> eftry.abc@gmail.com af@abc.com 	<ul style="list-style-type: none"> eft***@gmail.com af***@abc.com
Bank card number	Only the last four digits in a credit card number or deposit card number are displayed in plaintext. All the other digits are displayed as asterisks (*).	<ul style="list-style-type: none"> 1234576834509782 643257829145430986 	<ul style="list-style-type: none"> *****9782 *****0986
IP address or MAC address	Only the first segment in an IP address or a MAC address is displayed in plaintext. All the other characters are displayed as asterisks (*).	<ul style="list-style-type: none"> 192.000.0.0 ab:cd:11:a3:a0:50 	<ul style="list-style-type: none"> 192.***.*.* ab:*.*.*.*.*.*.*
License plate number	Only the one-character provincial abbreviation and the last three characters in a license plate number in mainland China are displayed in plaintext. All the other characters are displayed as asterisks (*).	<ul style="list-style-type: none"> (One-character provincial abbreviation)AP555B (One-character provincial abbreviation)ADP555T 	<ul style="list-style-type: none"> (One-character provincial abbreviation)A**55B (One-character provincial abbreviation)A***55T

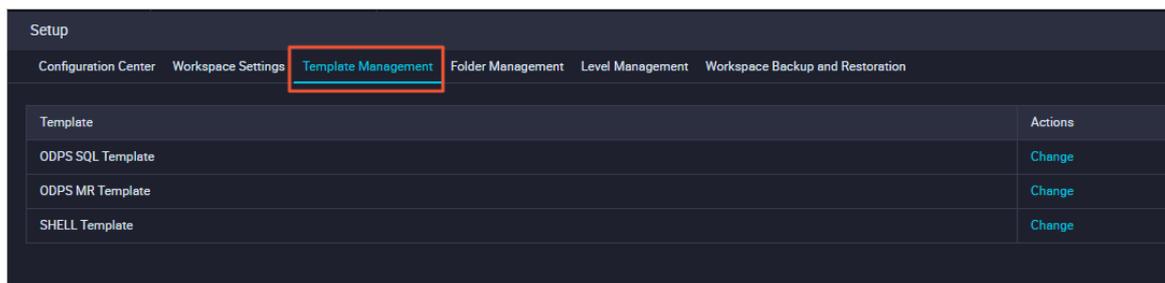
 **Note** If you want to de-identify more data types or have special requirements on the de-identified data formats, complete your de-identification settings in Data Security Guard. The feature of de-identifying data for DataWorks workspaces must work with Data Security Guard. For more information, see [Data security guard](#).

2.5.9.4. Template management

The Template Management page displays code templates. Workspace administrators can change the display formats of the templates as required.

Procedure

1. [Log on to the DataWorks console](#).
2. On the **Data Analytics** tab, click  in the lower-left corner.
3. On the **Setup** page, click the **Template Management** tab.



Note Templates are only available for ODPS SQL, ODPS MR, and Shell nodes.

4. Find the target template and click **Change** in the Actions column.
5. In the **Node Template** dialog box, enter the template as required.
6. Click **Save**.

2.5.9.5. Folder management

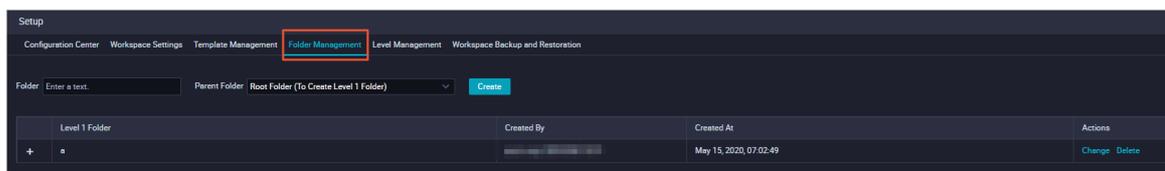
Each workspace can hold a great number of tables. For easy management, you can organize tables in two levels of folders.

Context

Folders are used to store tables. A workspace administrator can add multiple folders and classify tables by purpose and name.

Procedure

1. Log on to the DataWorks console.
2. On the **Data Analytics** tab, click  in the lower-left corner.
3. On the **Setup** page, click the **Folder Management** tab.



On the page that appears, you can add, modify, and delete folders.

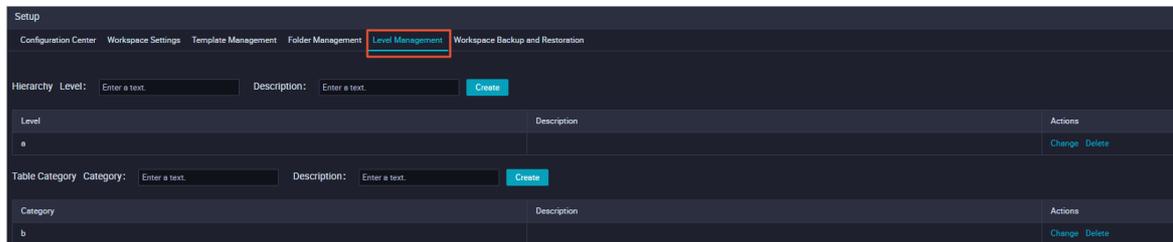
- o Add a folder
Enter a custom folder name in the **Folder** field, select a parent folder from the **Parent Folder** drop-down list, and then click **Create**.
- o Modify a folder
Find the target folder and click **Change** in the Actions column. In the **Change Folder** dialog box, enter a new folder name and click **OK**.
- o Delete a folder
Find the target folder and click **Delete** in the Actions column. In the **Delete Folder** message, click **OK**.

2.5.9.6. Level management

On the Level Management tab, you can design physical levels of tables.

Procedure

1. Log on to the DataWorks console.
2. On the Data Analytics tab, click  in the lower-left corner.
3. On the Setup page, click the **Level Management** tab.



You can classify tables based on their importance. Level management allows you to precisely locate incorrectly organized tables and ensures normal running of published jobs.

If a workspace does not contain default table levels, the workspace owner or workspace administrator must add them as required.

On the **Level Management** tab, you can add, modify, and delete table levels.

- o To add a table level, perform the following steps:
 - a. In the **Hierarchy** section, set **Level** and **Description**.
 - b. Click **Create**.
- o To modify a table level, perform the following steps:
 - a. Find the target table level and click **Change** in the **Actions** column.
 - b. In the **Change Level** dialog box, modify **Level** and **Description** as needed.
 - c. Click **OK**.
- o To delete a table level, perform the following steps:
 - a. Find the target table level and click **Delete** in the **Actions** column.
 - b. In the **Delete Level** message, click **OK**.

On the **Level Management** tab, you can also add, modify, and delete table categories.

- o To add a table category, perform the following steps:
 - a. In the **Table Category** section, set **Category** and **Description**.
 - b. Click **Create**.
- o To modify a table category, perform the following steps:
 - a. Find the target table category and click **Change** in the **Actions** column.
 - b. In the **Change Category** dialog box, modify **Category** and **Description** as needed.
 - c. Click **OK**.
- o To delete a table category, perform the following steps:
 - a. Find the target table category and click **Delete** in the **Actions** column.
 - b. In the **Delete Category** message, click **OK**.

2.5.9.7. Workspace backup and restore

On the Workspace Backup and Restoration tab, you can migrate code between workspaces. This topic describes how to back up and restore a workspace.

Prerequisites

Workspaces are created. For more information, see [Create a workspace](#).

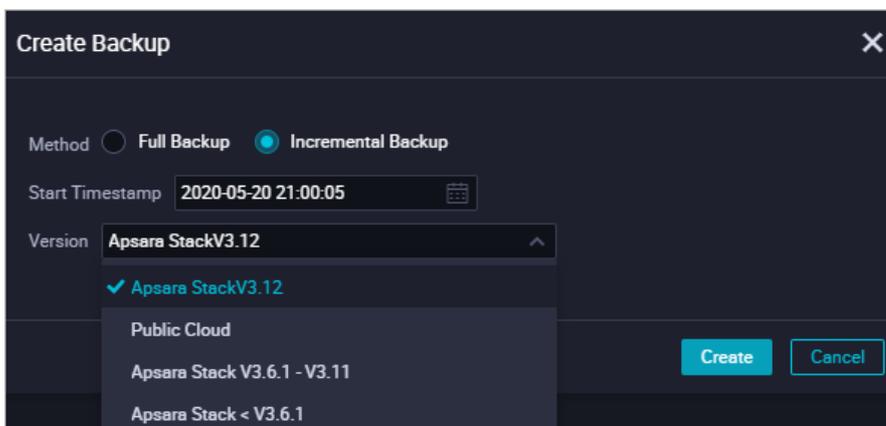
Go to the Workspace Backup and Restoration tab

1. [Log on to the DataWorks console](#).
2. On the **Data Analytics** tab, click  in the lower-left corner.
3. On the **Setup** page, click the **Workspace Backup and Restoration** tab. On the page that appears, you can back up and restore workspaces.
 - On the **Backup** tab, you can compress the node code, node dependencies, resources, and functions in a workspace into one package.
 - On the **Restore** tab, you can restore a workspace to its original scheduling settings. After the workspace is restored, all nodes in the workspace are saved but not committed.

Back up a workspace

A workspace backup is a compressed package containing the node code, node dependencies, resources, and functions in the workspace.

- Only workspace administrators can export backups and restore data from backups.
 - Workflows and node groups of earlier versions cannot be backed up. We recommend that you use the latest version for data analytics.
 - A node backed up to a path in the workspace will override the original node with the same name in the path. We recommend that you create another workspace to restore data.
 - Data in tables is not backed up when you back up a workspace. You can synchronize the table data in the following ways:
 - Click the **Workspace Manage** icon in the upper-right corner. On the page that appears, click **Data Source** and configure a MaxCompute connection. Then, create a sync node to back up the data.
 - In workspace A, run the data definition language (DDL) statement `create table select * from workspace B. Table name` to migrate data.
1. On the **Workspace Backup and Restoration** tab, click **Backup**.
 2. Click **Create Backup** in the upper-right corner.
 3. In the **Create Backup** dialog box, set the parameters as required.



Parameter	Description
-----------	-------------

Parameter	Description
Method	<p>The method used to back up the workspace. Valid values:</p> <ul style="list-style-type: none"> ◦ Full Backup: Back up all the node code, node dependencies, resources, and functions in the workspace. ◦ Incremental Backup: Back up all the new or modified nodes from the timestamp specified by the Start Timestamp parameter to the current time. <p>Note If you use the incremental backup method, make sure that the dependencies between incremental sync nodes are correct. Otherwise, the workspace may fail to be restored. We recommend that you set this parameter to Full Backup.</p>
Start Timestamp	The start time point at which data in the workspace is backed up. This parameter is available only when you set Method to Incremental Backup .
Version	The version of the workspace to be backed up. Valid values: Apsara Stack V3.12 , Public Cloud , Apsara Stack V3.6.1 - V3.11 , and Apsara Stack < V3.6.1 .

4. Click **Create**. After the data is backed up, click **Download** to download the backup data to a local device.

Restore a workspace

1. On the **Workspace Backup and Restoration** tab, click **Restore**.
2. Click **Restore** in the upper-right corner.
3. In the **Restore** dialog box, click **Select File**.

Note You can upload the compressed package that you previously backed up to the workspace.

4. Click **Restore**.
5. Click **Set Compute Engine Mapping**. In the dialog box that appears, set the mapping between the compute engines of the current workspace and the destination workspace.
6. Click **OK**. If the workspace that you backed up contains multiple compute engines, the system scans all compute engine instances during restoration. The system only restores nodes of the existing compute engines in the workspace to be restored. In this case, you must configure the mappings between the compute engines before restoring the destination workspace.

Note

- If the workspace to be restored does not contain a compute engine type such as E-MapReduce, or no instance is available for the compute engine type, nodes of this engine type are not restored.
- Compute engine mappings must be configured for custom node types.

2.5.10. Deploy

2.5.10.1. Deploy nodes

In a rigorous data development process, developers develop and debug code and configure dependencies and scheduling properties for nodes in the development environment. Then, developers commit and deploy the nodes to run them in the production environment.

DataWorks workspaces in standard mode can process data seamlessly from the development environment to the production environment within a single workspace. We recommend that you use workspaces in standard mode to develop and produce data.

Deploy nodes in a workspace in standard mode

Each DataWorks workspace in standard mode is linked with two MaxCompute projects, one as the development environment and the other as the production environment. You can directly commit and deploy nodes from the development environment to the production environment.

Follow these steps:

1. On the **DataStudio** page, configure and debug the code of nodes. Then, double-click the target workflow in the left-side navigation pane. On the dashboard of the workflow that appears, click the **Submit** icon to check whether the dependencies between nodes are correct and commit the nodes.
2. After the nodes are committed, click the **Deploy** icon.
3. On the **Create Deploy Task** page that appears, select the target nodes and click **Add to List**. The nodes are added to the to-be-deployed node list.

You can search for nodes by condition, such as the committer, node type, change type, time when a node is committed, node name, and node ID. If you click **Deploy Selected**, the selected nodes are deployed to the production environment.

4. Click **View List**. In the **Nodes to Deploy** dialog box that appears, click **Deploy All**. All nodes in the list are deployed to the production environment.

 **Note** Workspaces in standard mode protect tables in the production environment from being manipulated, and therefore provide the stable, secure, and reliable production environment. We recommend that you use workspaces in standard mode to deploy and run nodes.

Clone nodes between workspaces in basic mode

You cannot deploy nodes in workspaces in basic mode. If you want to isolate the development environment from the production environment for workspaces in basic mode, create two workspaces, one for development and the other for production. You can clone nodes from the development workspace to the production workspace.

As shown in the following figure, two workspaces in basic mode are created, one for development and the other for production. You can use the cross-workspace cloning feature to clone nodes from Workspace A to Workspace B, and then commit the cloned nodes to the scheduler for scheduling.

-  **Note**
- **Permission requirement:** Only workspace administrators and Resource Access Management (RAM) users who have the O&M permissions can clone nodes.
 - **Workspace type:** You can only clone nodes in workspaces in basic mode, but cannot clone those in workspaces in standard mode.
 - **Prerequisites:** The source workspace in basic mode and the destination workspace in basic mode are created.

1. Commit nodes.

After you create and configure nodes in the source workspace, commit the nodes on the dashboard of the target workflow.

2. Click **Cross-Workspace Cloning**.

3. On the Create Clone Task page that appears, select the target nodes and the destination workspace, and then click **Add to List**.

4. Clone the nodes. Click **View List**. In the To-Be-Cloned Nodes dialog box that appears, check the nodes to be cloned and click **Clone All**.

In the Create Clone Task dialog box that appears, click **Clone**.

5. View the cloned nodes.

You can view the successfully cloned nodes on the View Clone Tasks page of the source workspace.

Switch to the destination workspace. You can find that the nodes are cloned from the source workspace.

2.5.10.2. Overview of cross-workspace cloning

For workspaces under the same Apsara Stack tenant account, you can use the cross-workspace cloning feature to clone and deploy workflows across these workspaces. You can also use this feature to clone nodes, such as computing or sync nodes, across workspaces. This topic describes how to process the dependencies between nodes during cross-workspace cloning.

If you clone nodes across workspaces by using the **cross-workspace cloning** feature, DataWorks automatically changes the output names of the cloned nodes in the destination workspace to distinguish nodes in different workspaces under the same Apsara Stack tenant account. This allows you to successfully clone node dependencies.

 **Note** Cross-workspace cloning cannot be used to clone nodes across workspaces in different regions.

You can set the owner of cloned nodes in the destination workspace to **Default** or **Clone Task Creator**.

- If you clone nodes owned by the workspace administrator:

After the nodes are cloned to the destination workspace, their owner is set to the original owner preferentially. If the original owner is not added to the destination workspace, you will become the owner.

- If you clone nodes owned by yourself:

After the nodes are cloned to the destination workspace, their owner is set to you preferentially. If you are not added to the destination workspace, you are asked whether to change the owner. If you agree to change the owner, you will be added to the destination workspace and become the owner of the cloned nodes. If you do not agree to change the owner, the clone task is canceled.

Clone a workflow

Assume that the output of the task_A node in the project_1 workspace is project_1.task_A_out. If you clone a workflow that contains the task_A node to the destination workspace project_2, the node output name changes to project_2.task_A_out in the destination workspace.

Clone node dependencies

Assume that the task_B node in the project_1 workspace depends on the task_A node in the project_3 workspace. If you clone the task_B node in the project_1 workspace to the destination workspace project_2, the dependency between the task_A and task_B nodes is also cloned. The task_B node in the project_2 workspace also depends on the task_A node in the project_3 workspace.

2.5.10.3. Clone nodes across workspaces

This topic describes how to clone nodes across workspaces with an example of cloning a workflow from one workspace to another.

Prerequisites

Two workspaces named `Weisong_dataworks_test` and `Weisong_dataworks_test2` respectively are created. For more information about how to create a workspace, see [Create a workspace](#).

Context

You can clone nodes across workspaces in the following scenarios:

- Clone nodes from a workspace in the basic mode to another workspace in the basic mode.
- Clone nodes from a workspace in the basic mode to another workspace in the standard mode.

After you clone a node, the folder and workflow to which the node belongs are cloned to the destination workspace. Any change to the node, folder, or workflow can also be cloned to the destination workspace.

Procedure

1. Log on to the DataWorks console. On the DataStudio page that appears, switch to the `Weisong_dataworks_test` workspace in the top navigation bar.
2. Select the target workflow. In the Data Analytics section, double-click the target workflow. On the workflow configuration page that appears, click **Cross-Workspace Cloning** in the upper-right corner. The Create Clone Task page appears.
3. Select the destination workspace, node type, and change type. On the **Create Clone Task** page, set **Target Workspace** to `Weisong_dataworks_test2`. Select the node type and change type of the node for the clone task as required, and select one or more target nodes that appear in the list. Then, click **Clone Selected**.
4. In the **Create Clone Task** dialog box that appears, check the destination workspace, target node, and change type, and then click **Clone**.
5. After the system message that indicates the target node is cloned successfully and is being committed to the destination workspace appears, switch to the `Weisong_dataworks_test2` workspace. In the **Data Analytics** section, view the workflow that has been successfully cloned to the current workspace.

2.5.11. Create an ad hoc query node

The Ad-Hoc Query tab allows you to test your code in the development environment. You can check for errors and check whether your code works as expected.

Context

You do not need to commit and deploy ad hoc query nodes or configure scheduling policies for ad hoc query nodes. You can configure scheduling policies only for nodes created under **Business Flow** on the **Data Analytics** tab.

Create a folder

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Ad-Hoc Query** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.
3. On the Ad-Hoc Query tab, move the pointer over  and select **Folder**.

4. In the **Create Folder** dialog box, set **Folder Name** and **Location**.

 **Note**

- The folder name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.). It is not case-sensitive.
- DataWorks supports multi-level folders. You can save a newly created folder under another folder that already exists.

5. Click **Commit**.

Create an ad hoc query node

You can create **ODPS SQL** and **Shell** nodes on the **Ad-Hoc Query** tab. This topic describes how to create an ODPS SQL node.

1. On the **Ad-Hoc Query** tab, right-click the target folder and choose **Create Node > ODPS SQL**.
2. In the **Create Node** dialog box, set **Node Name** and **Location**.

 **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.). It is not case-sensitive.

3. Click **Commit**.
4. On the node configuration tab that appears, enter an SQL statement.
5. Click  in the toolbar.

2.5.12. View runtime logs

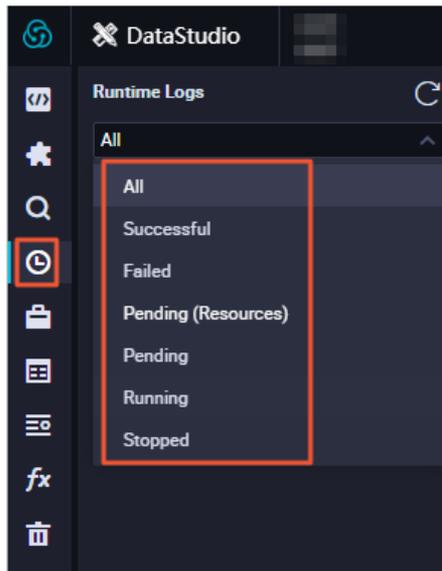
The **Runtime Logs** tab displays the records of all nodes that have been run in the last three days. You can click a node to view its runtime logs.

Context

The runtime logs are retained for only three days.

Procedure

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Runtime Logs** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.
3. Select a node state from the drop-down list to view the runtime logs of nodes in the specified state.



4. Click a record to view the runtime log on the right. If you need to save the SQL statements in the runtime log, click  in the toolbar. In the **Create Node** dialog box, set the parameters and click **Commit** to save the SQL statements that have been run as an ad hoc query node.

2.5.13. View tenant tables

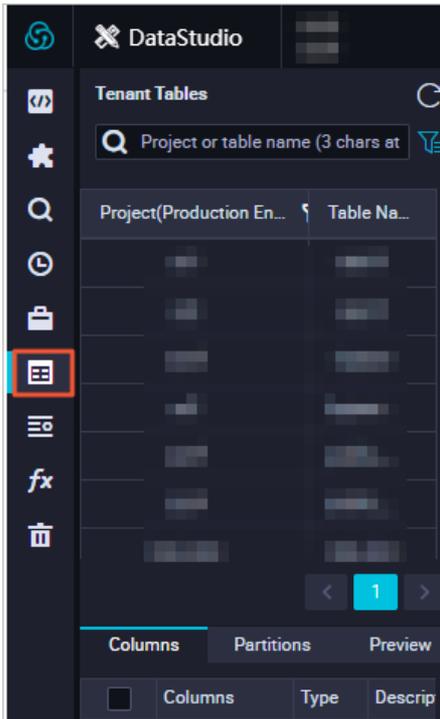
On the Tenant Tables tab, you can view tables of all workspaces of the current tenant account.

Prerequisites

The **Tenant Tables** tab appears only after you bind a MaxCompute compute engine on the **Project Management** page. For more information, see [Configure a workspace](#).

Procedure

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Tenant Tables** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.
3. View MaxCompute tenant tables.



Parameter or tab	Description
Project Name	<p>The name of the workspace in the corresponding environment.</p> <p>Click  next to the search box and select the target environment to switch to the environment.</p> <div style="background-color: #e0f2f1; padding: 10px;"> <p> Note</p> <ul style="list-style-type: none"> ◦ For a workspace in standard mode, the Tenant Tables tab displays tables in both the development environment and the production environment. For a workspace in basic mode, the Tenant Tables tab displays only the tables in the production environment. ◦ The current environment is marked in blue. </div>
Table Name	The name of the table in the corresponding workspace.
Columns tab	Displays the name, data type, and description of fields in the table.
Partitions tab	<p>Displays the partition information of the current table. A maximum of 60,000 partitions are supported. If you have specified the TTL for partitions, the number of partitions depends on the TTL.</p> <div style="background-color: #e0f2f1; padding: 10px;"> <p> Notice The partition information is displayed only for MaxCompute tenant tables.</p> </div>

Parameter or tab	Description
Preview tab	<p>Displays the data of the current table.</p> <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px;"> <p> Notice You can preview only the data of MaxCompute tenant tables.</p> </div>

2.5.14. Manage tables

This topic describes how to view, modify, and delete MaxCompute tables, and the basic knowledge about data hierarchy.

Prerequisites

The **Workspace Tables** tab appears only after you bind a MaxCompute compute engine on the **Project Management** page. For more information, see [Configure a workspace](#).

Manage tables

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Workspace Tables** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.
3. View and manage tables. The following section describes how to view, modify, and delete a MaxCompute table. For more information about how to create a table, see [Create a MaxCompute table](#).

Operation	Description
View a table	<p>Click  next to the search box and select the target environment to switch to the environment.</p> <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px;"> <p> Note</p> <ul style="list-style-type: none"> ○ For a workspace in standard mode, the Workspace Tables tab displays tables in both the development environment and the production environment. ○ For a workspace in basic mode, the Workspace Tables tab displays only the tables in the production environment. ○ The current environment is marked in blue. </div> <p>Double-click a table to view its details on the table configuration tab.</p>
Import data to a table	<p>On the Workspace Tables tab, click  to import data to a table. For more information, see Create tables and import data.</p>

Divide a data warehouse into layers

In the **Physical Model** section of the configuration tab of a table, you can define table layers for a data warehouse. This allows you to have better planning and control over your data.

Typically, a data warehouse consists of the following layers:

- ODS: The ODS layer stores raw data of the source system based on the original data structure. The ODS

layer serves as the data staging area of the data warehouse. It imports basic data to MaxCompute and records historical changes of basic data.

- CDM: The CDM layer consists of the dimension data (DIM), data warehouse detail (DWD), and data warehouse service (DWS) layers. The CDM layer processes and integrates the data of the ODS layer to define conformed dimensions, create reusable detailed fact tables for data analysis and statistics collection, and aggregate common metrics.
 - The DIM layer defines conformed dimensions for an enterprise based on the concepts of dimensional modeling. It reduces the risk of inconsistent statistical criteria and algorithms.

Tables at the DIM layer are also called logical dimension tables. Generally, each dimension corresponds to a logical dimension table.
 - The DWS layer is driven by analyzed subjects during data modeling. Based on the metric requirements of upper-layer applications and products, the DWS layer creates fact tables to aggregate common metrics and builds a physical data model by using wide tables. The DWS layer creates statistical metrics in compliance with uniform naming conventions and statistical criteria, provides common metrics for the upper layer, and generates aggregate wide tables and detailed fact tables.

Tables at the DWS layer are also called logical aggregate tables, which are used to store derived metrics.
 - The DWD layer is driven by business processes during data modeling. It creates detailed fact tables at the finest granularity based on each specific business process. In combination with the data usage habits of an enterprise, you can duplicate some key attribute fields of dimensions in detailed fact tables to create wide tables.

Tables at the DWD layer are also called logical fact tables.
- ADS: The ADS layer stores personalized statistical metrics of data products. It processes the data of the CDM and ODS layers.

2.5.15. View built-in functions

The Built-In Functions tab displays functions built in MaxCompute. You can view the types, description, and examples of functions on this tab.

Procedure

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Built-In Functions** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.
3. View the types, description, and examples of the built-in functions. Functions are categorized into aggregate functions, analytic functions, date functions, mathematical functions, string functions, and other functions. The preceding functions are built in MaxCompute. You can click a function to view its description.

2.5.16. Manage deleted nodes

DataWorks provides a recycle bin to store all deleted nodes in the current workspace. You can restore or permanently delete the nodes.

Go to the Recycle Bin tab

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Recycle Bin** icon. Click  in the lower-left corner to show or hide the left-side navigation pane.

3. View all the deleted nodes in the current workspace. On this tab, you can delete or restore a deleted node.

 **Notice**

- The recycle bin displays only 100 nodes. If more than 100 nodes are deleted, the nodes deleted earlier are deleted permanently from the recycle bin.
- Deleted node groups are not displayed in the recycle bin.

Restore a node in the recycle bin

1. On the **Recycle Bin** tab, right-click a deleted node.
2. Select **Restore**.
3. In the **Restore Node** message, click **OK**.

 **Note** After you restore a node, a new node ID is generated for scheduling and all the information about the node is restored.

Permanently delete a node from the recycle bin

1. On the **Recycle Bin** tab, right-click a deleted node.
2. Select **Delete**.
3. In the **Delete** message, click **OK**.

 **Note** Nodes permanently deleted from the recycle bin cannot be restored. The recycle bin displays only deleted nodes.

2.5.17. Create a manually triggered workflow

In a manually triggered workflow, all nodes must be manually triggered, and cannot be automatically scheduled by DataWorks. Therefore, you do not need to specify parent nodes or outputs for nodes in manually triggered workflows.

Create a manually triggered workflow

1. Log on to the DataWorks console.
2. On the left-side navigation submenu, click the **Manually Triggered Workflows** icon.

Click  in the lower-left corner to show or hide the left-side navigation pane.

3. Right-click **Manually Triggered Workflows** and select **Create Workflow**.
4. In the **Create Workflow** dialog box, set **Workflow Name** and **Description**.

 **Notice** The workflow name must be 1 to 128 characters in length and can contain letters, digits, underscores (`_`), and periods (`.`). It is not case-sensitive.

5. Click **Create**.

Composition of a manually triggered workflow

 **Note** We recommend that you create a maximum of 100 nodes in a manually triggered workflow.

A manually triggered workflow consists of the nodes of the following modules. After you create a manually triggered workflow, open this workflow and create nodes of various types for each module. For more information, see [Node types](#).

- **Data Integration**

Double-click **Data Integration** under the created workflow to view all the data integration nodes.

Right-click **Data Integration** and choose **Create > Batch Synchronization** to create a batch sync node. For more information, see [Create a batch sync node](#).

- **MaxCompute**

The MaxCompute compute engine consists of data analytics nodes, such as ODPS SQL, SQL Snippet, ODPS Spark, PyODPS, ODPS Script, and ODPS MR nodes. You can also view and create tables, resources, and functions.

- **Data Analytics**

Show **MaxCompute** under the created workflow and right-click **Data Analytics** to create a data analytics node. For more information, see [Create an ODPS SQL node](#), [Create an SQL Snippet node](#), [Create an ODPS Spark node](#), [Create a PyODPS node](#), [Create an ODPS Script node](#), and [Create an ODPS MR node](#).

- **Table**

Show **MaxCompute** under the created workflow and right-click **Table** to create a table. You can also view all the tables created for the current MaxCompute compute engine. For more information, see [Create a MaxCompute table](#).

- **Resource**

Show **MaxCompute** under the created workflow and right-click **Resource** to create a resource. You can also view all the resources created for the current MaxCompute compute engine. For more information, see [Create, reference, and download resources](#).

- **Function**

Show **MaxCompute** under the created workflow and right-click **Function** to create a function. You can also view all the functions created for the current MaxCompute compute engine. For more information, see [Register a UDF](#).

- **Algorithm**

Click the created workflow and right-click **Algorithm** to create an algorithm. You can also view all the PAI nodes created in the current manually triggered workflow. For more information, see [Create a PAI node](#).

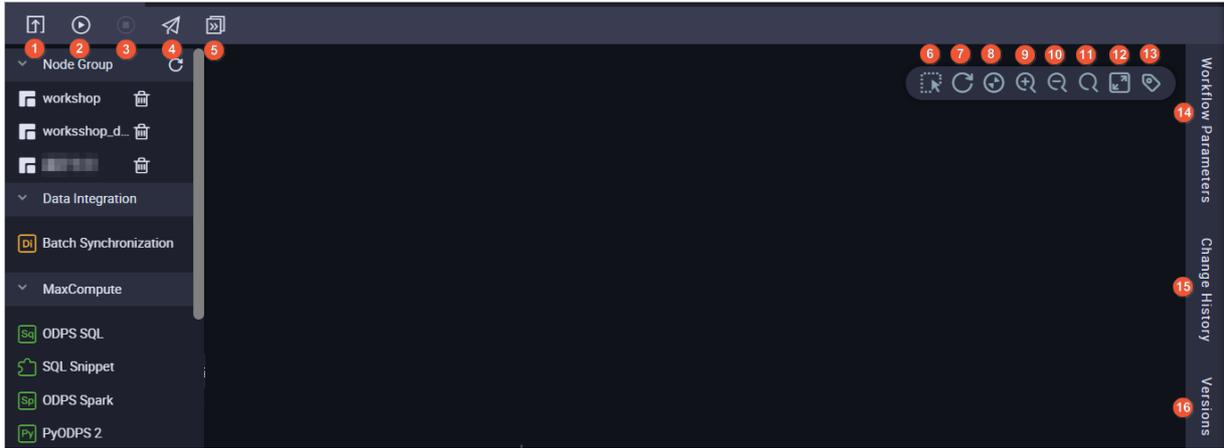
- **General**

Click the created workflow and right-click **General** to create relevant nodes. For more information, see [Create a Shell node](#) and [Create a zero-load node](#).

- **UserDefined**

Click the created workflow and right-click **UserDefined** to create relevant nodes. For more information, see [Create a Hologres development node](#).

GUI elements



The following table describes the icons and tabs on the Manually Triggered Workflows page.

No.	Icon or tab	Description
1	Submit icon	Commits all nodes in the current manually triggered workflow.
2	Run icon	Runs all nodes in the current manually triggered workflow. Nodes in this workflow do not have dependencies, and therefore they can run at a time.
3	Stop icon	Stops all running nodes in the current manually triggered workflow.
4	Deploy icon	Navigates to the Deploy page. On this page, you can deploy some or all nodes that are committed but not deployed to the production environment. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p>Note This icon is available only when the workspace is in standard mode.</p> </div>
5	Go to Operation Center icon	Navigates to the Operation Center page.
6	Box. icon	Box-selects a node group consisting of required nodes.
7	Refresh icon	Refreshes the page of the current manually triggered workflow.
8	Auto Layout icon	Sorts the nodes in the current manually triggered workflow.
9	Zoom In icon	Zooms in the current page.
10	Zoom Out icon	Zooms out the current page.
11	Search icon	Searches for a node in the current manually triggered workflow.
12	Toggle Full Screen View icon	Displays nodes in the current manually triggered workflow in the full screen.
13	Show Engine Information/Hide Engine Information icon	Shows or hides engine information.

No.	Icon or tab	Description
14	Workflow Parameters tab	Allows you to set parameters. Parameters set on this tab have a higher priority than those specified on the corresponding node configuration tab. If two values are set separately, the value set on the Workflow Parameters tab takes effect.
15	Change History tab	Allows you to view the operation records of all nodes in the current manually triggered workflow.
16	Versions tab	Allows you to view the deployment records of all nodes in the current manually triggered workflow.

2.5.18. Editor keyboard shortcuts

This section describes keyboard shortcuts available for the code editor.

Google Chrome in Windows OS

Ctrl + S : Save changes to a node.

Ctrl + Z : Undo an action.

Ctrl + Y : Redo an action.

Ctrl + D : Select occurrences.

Ctrl + X : Cut a line.

Ctrl + Shift + K : Delete a line.

Ctrl + C : Copy a line.

Ctrl + I : Select a line.

Alt + Shift + Drag : Select a block.

Alt + Click : Insert an additional cursor.

Ctrl + Shift + L : Select all occurrences.

Ctrl + F : Search for text in a node.

Ctrl + H : Replace text in a node.

Ctrl + G : Locate a line.

Alt + Enter : Select all matched strings.

Alt + Up or down arrow : Move a line up or down.

Alt + Shift + Up or down arrow : Duplicate a line.

Ctrl + Shift + K : Delete a line.

Ctrl + Enter or Ctrl + Shift + Enter : Insert a line break downwards or upwards.

Ctrl + Shift + Back slash (\) : Jump to the parenthesis, bracket, or brace that matches the adjacent one.

Ctrl + Left bracket ([) or right bracket (]) : Increase or decrease the indent of a line.

Home or End : Move the cursor to the beginning or end of a line.

Ctrl + Home or End : Move the cursor to the top or bottom of a node.

Ctrl + Left or Right arrow : Move the cursor one word to left or right.

Ctrl + Shift + Left bracket (]) or right bracket ([) : Hide or show a block.

Ctrl + K + Left bracket (]) or right bracket ([) : Hide or show sub-blocks in a block.

Ctrl + K + 0 or J : Hide or show all blocks.

Ctrl + Slash (/) : Comment out or uncomment the selected lines or blocks.

Google Chrome in Mac OS

Command-S : Save changes to a node.

Command-Z : Undo an action.

Command-Y : Redo an action.

Command-D : Select occurrences.

Command-X : Cut a line.

Shift-Command-K : Delete a line.

Command-C : Copy a line.

Command-I : Select a line.

Command-F : Search for text in a node.

Option-Command-F : Replace text in a node.

Option-Up or down arrow : Move a line up or down.

Option-Shift-Up or down arrow : Duplicate a line.

Shift-Command-K : Delete a line.

Command-Enter or Shift-Command-Enter : Insert a line break downwards or upwards.

Shift-Command-Back slash (\) : Jump to the parenthesis, bracket, or brace that matches the adjacent one.

Command-Left bracket (]) or right bracket ([) : Increase or decrease the indent of a line.

Command-Left or right arrow : Move the cursor to the beginning or end of a line.

Command-Up or down arrow : Move the cursor to the top or bottom of a node.

Option-Left or right arrow : Move the cursor one word to left or right.

Option-Command-Left bracket (]) or right bracket ([) : Hide or show a block.

Command-K-Left bracket (]) or right bracket ([) : Hide or show sub-blocks in a block.

Command-K-0 or J : Hide or show all blocks.

Command-Slash (/) : Comment out or uncomment the selected lines or blocks.

Insert multiple cursors and select multiple occurrences or lines

Option-Click : Insert an additional cursor.

Option-Command-Up or down arrow : Insert an additional cursor to the previous or next line.

Command-U : Undo a cursor-related operation.

Option-Shift-I : Insert a cursor at the end of each selected line.

Command-G or Shift-Command-G : Select the next or previous matched string.

Command-F2 : Select the nearest character of each cursor.

Shift-Command-L : Select the nearest word of each cursor.

Option-Enter : Select all the matched strings.

Option-Shift-Drag : Multi-select lines

Option-Shift-Command-Up or down arrow : Extend a selection one line up or down.

Option-Shift-Command-Left or right arrow : Extend a selection one character to the left or right.

2.5.19. Use E-MapReduce in DataWorks

This topic describes how to use E-MapReduce in DataWorks.

Bind an E-MapReduce project to a DataWorks workspace

Note Before you bind an E-MapReduce project to a DataWorks workspace, you must obtain the information about the E-MapReduce project.

1. Log on to the DataWorks console.
2. On the DataStudio page, click  in the upper-right corner. The **Project Management** page appears.
3. Click the **E-MapReduce** tab in the **Computing Engine Information** section. On this tab, you can view the information about all available E-MapReduce compute engines in the current workspace.
4. Click **Add instances**.
5. In the **New EMR cluster** dialog box, set the parameters as required.

Parameter	Description
Instance display name	The name of the E-MapReduce compute engine instance.
Region	The region of the workspace.
Access ID	The AccessKey ID of the account authorized to access the E-MapReduce cluster.
Access Key	The AccessKey secret of the account authorized to access the E-MapReduce cluster.

Parameter	Description
EmrClusterID	The ID of the E-MapReduce cluster.
Cluster ID	The ID of the user who created the E-MapReduce cluster.
Project ID	The ID of the project in the E-MapReduce cluster.
YARN resource queue	The name of the resource queue in the E-MapReduce cluster. Unless otherwise specified, set the value to <i>default</i> .
Endpoint	The endpoint of the E-MapReduce cluster. You can obtain the endpoint in the Apsara Stack Operations console. For more information, see Obtain an endpoint .

6. Click **Confirm**. After the E-MapReduce cluster is bound to your workspace, you can create E-MapReduce nodes on the **DataStudio** page.

-  **Note** If the binding fails, check whether the failure is caused by one of the following reasons:
- The E-MapReduce user ID is bound to another tenant account.
 - The specified cluster name already exists.

Create an E-MapReduce node

E-MapReduce nodes are categorized into four types: EMR Hive, EMR Spark SQL, EMR Spark, and EMR MR. For more information, see [Create an EMR MR node](#).

Reference resource files

E-MapReduce resource files are categorized into two resource types: EMR JAR and EMR File.

Reference E-MapReduce resource files by using the following methods:

- For EMR Hive and EMR MR nodes, add `--@resource_reference{"Resource name"}` at the first line of the code.
- For EMR Spark nodes, add `##@resource_reference{"Resource name"}` at the first line of the code.

Manage data

DataWorks allows you to query E-MapReduce metadata and synchronize the data for data development.

2.6. HoloStudio

2.6.1. Overview

This topic describes what is HoloStudio and the features of HoloStudio.

HoloStudio, which is developed based on Hologres, is an all-in-one online analytical processing (OLAP) development platform deeply integrated with DataWorks. HoloStudio provides Hologres users with standardized and easy-to-use development services and one-stop real-time data warehouse building services by using a visualized and wizard-based user interface. In addition to standard management available in PostgreSQL, HoloStudio provides more interactive analytics features. HoloStudio supports the following features:

- Table management

HoloStudio allows you to create a PostgreSQL table on a visualized user interface or by using SQL statements. HoloStudio also allows you to create a foreign table sourced from MaxCompute by synchronize the schema with one click and preview and analyze MaxCompute data.

- **SQL Console**

The SQL Console module of HoloStudio allows you to use an SQL editor to obtain query results in seconds.

- **Data analytics**

Based on the underlying capabilities of DataWorks, HoloStudio allows you to create multiple foreign tables at a time and synchronize data with one click and provides you with the one-stop, stable, and efficient extract-transform-load (ETL) service.

- **Seamless connection to the Hologres console**

HoloStudio is seamlessly connected to the Hologres console. In the Hologres console, you can manage Hologres instances, users, and databases on a visualized user interface.

2.6.2. Bind a Hologres database to the current workspace

This topic describes how to bind a Hologres database to the current workspace.

If you need to use HoloStudio for data development, bind a Hologres database to the current workspace. Perform the following steps:

1. Log on to HoloStudio.

On the left-side navigation submenu, click **PG management**. On the PG management tab, move the pointer over the Create icon and select **Database**.

2. In the Create Database dialog box, set relevant parameters.

In the **Create Database** dialog box, set relevant parameters and click **Test connectivity**. If the system message indicating that the connectivity test is passed appears, the specified database is connected. Then, click **Complete**.

Parameter	Description	Remarks
Connect To	The type of the data store.	The value is automatically generated and cannot be changed.
Server	The endpoint of the Hologres instance.	You can view the endpoint on the Basic Information page of the Hologres instance in the Hologres console.
Port	The port number of the Hologres instance.	You can view the port number on the Basic Information page of the Hologres instance in the Hologres console.
Database Name	The name of the Hologres database to be bound to the current workspace.	In actual scenarios, create a Hologres database and set this parameter to the name of the database to bind the database to the current workspace.

Parameter	Description	Remarks
User name	The AccessKey ID of your Apsara Stack tenant account.	You can click the username in the upper-right corner to view the AccessKey ID.
Password	The AccessKey secret of your Apsara Stack tenant account.	You can click the username in the upper-right corner to view the AccessKey secret.
JDBC Extension	The extension parameters used to establish a Java Database Connectivity (JDBC) connection to Hologres.	Example: <code>?preferOuervMode=simple&tcpKeepAlive=true</code>
Test connectivity	Tests whether the database is connected.	N/A

3. Use the Hologres database in HoloStudio.

After the Hologres database is bound to the current workspace, click the Refresh icon on the **PG management** tab. After the database appears, you can use the database in HoloStudio.

2.6.3. SQL Console

The SQL Console in HoloStudio is an editor for executing SQL statements. In the SQL Console, you can execute SQL statements to analyze data in Hologres and obtain the query results. This topic describes the basic features and usage of the SQL Console in HoloStudio.

Folder

The Folder module stores new ad hoc queries, which helps you manage ad hoc queries.

In the left-side navigation pane, click **SQL Console**. Move the pointer over the Create icon and select **Folder**. In the Create Folder dialog box, enter a folder name and click Commit to create a folder. You can create an ad hoc query in the folder and execute standard SQL statements on tables. You can also right-click a table in the folder and select the relevant menu item to move, rename, or delete the table.

SQL Console

The SQL Console module allows you to create ad hoc queries and execute standard SQL statements.

1. Create an ad hoc query.

In the left-side navigation pane, click **SQL Console**. Move the pointer over the Create icon and select **SQL Console**. In the Create Node dialog box, set relevant parameters.

The following table describes the parameters for creating an ad hoc query.

Parameter	Description
Node Name	The name of the ad hoc query. The name can contain letters, digits, underscores (_), and periods (.).
Location	The folder where the ad hoc query is stored.
Database	The target database in which the ad hoc query is run.

2. Run the ad hoc query.

Write the SQL statements used in the ad hoc query and click the **Run** icon to run the query. Then, you can check the query result. The following example shows how to create a table, import data to the table, and then query the table:

```
CREATE TABLE supplier (
  s_suppkey bigint NOT NULL,
  s_name text NOT NULL,
  s_address text NOT NULL,
  s_nationkey bigint NOT NULL,
  s_phone text NOT NULL,
  s_acctbal bigint NOT NULL,
  s_comment text NOT NULL,
  PRIMARY KEY (s_suppkey)
);
INSERT INTO supplier VALUES
(1, 'Supplier#000000001', 'gf0JBoQDd7tgrzrddZ', 17, '27-918-335-1736', 575594, 'each slyly above the car
eful'),
(6, 'Supplier#000000006', 'tQxuVm7s7CnK', 14, '24-696-997-4969', 136579, 'final accounts. regular dolph
ins use against the furiously ironic decoys. '),
(10, 'Supplier#000000010', 'Saygah3gYWmp72i PY', 24, '34-852-489-8585', 389191, 'ing waters. regular request
s ar'),
(18, 'Supplier#000000001', 'PGGVE5PWAMwKDZw', 16, '26-729-551-1115', 704082, 'accounts snooze slyly furiou
sly bold'),
(39, 'Supplier#000000039', 'SYpEPWr1yAFHaC91qjFcijjeU5eH', 8, '18-851-856-5633 611565', 88990, 'le slyly re
quests. special packages shall are blithely. slyly unusual packages sleep'),
(48, 'Supplier#000000048', 'FNPMQDuyuKvTnLXXaLf3Wl6OtONA6mQlWJ', 14, '24-722-551-9498', 563062, 'xp
ress instructions affix. fluffily even requests boos');
SELECT * FROM supplier;
```

The following table describes the GUI elements on the node editing tab.

GUI element	Description
SQL editor	You can write SQL statements in the SQL editor.
Save icon	You can click this icon to save all statements in the SQL editor.
Run icon	You can click this icon to execute all statements in the SQL editor. The result appears on the Result tab. You can also select an SQL statement to be executed. In this case, the system only executes this statement.
Refresh icon	You can click this icon to refresh the content in the SQL editor. The system retains only the saved content after the refresh.
Stop icon	You can click this icon to stop executing SQL statements.
Runtime Log	You can check the execution results and, if any, error messages.
Result	You can check the table content after the SQL statements are executed.

GUI element	Description
Steal Lock icon	You can click this icon to unlock the locked SQL Console.

HoloStudio also allows you to directly manage the queried data. For example, you can hide columns, copy data, and search for data.

 **Note** For statements without results returned, such as the `CREATE TABLE` statement, only operational logs are generated after the statements are executed.

2.6.4. PostgreSQL management

2.6.4.1. Manage databases

The PG management module of HoloStudio allows you to manage databases and tables with one click in a visualized manner. In addition, HoloStudio supports interactive queries with a response time within seconds. This topic describes how to use the PG management module to manage databases.

Create a database

The PG management module allows you to connect databases to HoloStudio and manage the connected databases in a visualized manner. Before you connect a database to HoloStudio, create the database in the Hologres console or by executing SQL statements in the SQL Console. The following example demonstrates how to create a database by executing SQL statements in the SQL Console and connect the database to HoloStudio:

1. Create an ad hoc query.

Log on to the DataWorks console as a superuser and go to the HoloStudio page. In the SQL Console, execute the following SQL statement:

```
create database dbname;
create database testdb; // Create a database named testdb.
```

2. Bind the Hologres database to the current workspace.

On the HoloStudio page, click **PG management** on the left-side navigation submenu. On the PG management tab, move the pointer over the Create icon and select **Database**.

3. In the Create Database dialog box, set relevant parameters.

In the **Create Database** dialog box, set relevant parameters and click **Test connectivity**. If the system message indicating that the connectivity test is passed appears, the specified database is connected. Then, click **Complete**.

Parameter	Description	Remarks
Connect To	The type of the data store. Default value: Hologres.	The value is automatically generated and cannot be changed.
Server	The endpoint of the Hologres instance.	You can view the endpoint on the Basic Information page of the Hologres instance in the Hologres console.

Parameter	Description	Remarks
Port	The port number of the Hologres instance.	You can view the port number on the Basic Information page of the Hologres instance in the Hologres console.
Database Name	The name of the database to be bound to the current workspace.	The value must be the same as the database name in the CREATE DATABASE statement, for example, <code>testdb</code> .
User name	The AccessKey ID of your Apsara Stack tenant account.	You can click the username in the upper-right corner to view the AccessKey ID.
Password	The AccessKey secret of your Apsara Stack tenant account.	You can click the username in the upper-right corner to view the AccessKey secret.
JDBC Extension	The extension parameters used to establish a JDBC connection to Hologres.	Example: <code>?preferQueryMode=simple&tcpKeepAlive=true</code>
Test connectivity	Tests whether the database is connected.	If the system message indicating that the connectivity test is passed appears, the specified database is connected.

Delete a database

The PG management module of HoloStudio allows you to delete databases. On the left-side navigation submenu, click **PG management**. On the PG management tab, right-click the database you want to delete and select **Delete Database**.

In the Delete Database message, click **Ok** to delete the database.

 **Note** Only a superuser of a database or the database owner configured by a superuser can delete the database.

View database details

On the left-side navigation submenu, click **PG management**. On the PG management tab, right-click the database for which you want to view details and select **Database details**.

2.6.4.2. Manage tables

Similar to PostgreSQL, Hologres manages data by using tables. The PG management module of HoloStudio allows you to manage tables in a visualized manner. You can create, check, or delete a table with one click. This topic describes how to use the PG management module of HoloStudio to manage tables.

Create a table

1. Create a table.

On the homepage of HoloStudio, click **Create Table**. Alternatively, click **PG management** on the left-side navigation submenu. On the PG management tab, move the pointer over the Create icon and select **Table**.

2. Edit the table content and attributes.

On the tab that appears, edit the table content and attributes, and click **Commit**. The following figure shows an example of a column-oriented table with primary keys.

Section or tab	Parameter	Description
General	Interactive Analytics Database	The database where the table resides.
	Table Name	The name of the table.
	Description	The description of the table.
Field	Field Name	The name of the field in the table.
	Data Type	The data type of the field.
	Primary Key Field	Specifies whether to use the field as the primary key for the table.
	Optional	Specifies whether the field can be null.
	Array	Specifies whether the field is an ordered array of elements.
	Description	The description of the field.
	Actions	The actions that you can perform on the field. For example, you can delete the field from the table, or move up or down the position of the field in the table.
Properties	Storage Mode	The storage mode of the table. Valid values: Row Store and Column Store. Default value: Column Store.
	Lifecycle (Seconds)	The lifecycle of the table. Default value: Permanent.
	Clustered Index	The index used for sorting.
	Dictionary Code Columns	The column based on whose values a dictionary mapping is built.
	Bitmap Column	The column on which bit code is built.
Partitioned Table	PARTITION BY LIST	The partition field.

Check a table

1. View the DDL statement used to create the table.

On the left-side navigation submenu, click **PG management**. Double-click the table you want to check and click **Generate DDL Statement** to check the SQL statement used to create the table.

2. Preview data.

On the left-side navigation submenu, click **PG management**. Double-click the table you want to check and click **Data Preview** to check the content of the table. If the table contains no data, you can only view the fields of the table.

Delete a table

On the left-side navigation submenu, click **PG management**. On the PG management tab, right-click the table you want to delete and select **Delete Table**.

In the Delete message, click **Ok**.

2.6.4.3. Manage foreign tables

In Hologres, a foreign table does not store data but maps the table from the external data source. The PG management module of HoloStudio allows you to create, query, or delete foreign tables. You can only analyze foreign tables sourced from MaxCompute. This helps you obtain the query results.

This topic describes how to use the PG management module of HoloStudio to manage foreign tables.

Create a foreign table

On the left-side navigation submenu, click **PG management**. On the PG management tab, move the pointer over the Create icon and select **External Table**. On the tab that appears, set parameters for creating a foreign table and click **Commit**. An existing MaxCompute table is used in the following example. After you search for a MaxCompute table by entering its name, HoloStudio automatically generates a foreign table based on the fields of the MaxCompute table after you click **Commit**.

Note

1. Before you create a foreign table in Hologres, make sure that its source table exists in a MaxCompute project.
2. The fields of a foreign table in Hologres have a one-to-one mapping with those of the source table in MaxCompute. You can query specific fields or all fields.

Section or icon	Parameter	Description
General	Interactive Analytics Database	The database where the foreign table to be created resides.
	Table Name	The name of the foreign table.
External Service	Types	The service type of the external table. You can only set this parameter to MaxCompute.
Table	Table	The source table in MaxCompute to be mapped.
Commit	Commit	You can click this button to commit the foreign table that you create.

Check a foreign table

1. Preview data.

On the left-side navigation submenu, click **PG management**. On the PG management tab, double-click the foreign table you want to check, and click **Data Preview** to check the content of the foreign table.

2. View the DDL statement used to create the table.

On the left-side navigation submenu, click **PG management**. On the PG management tab, double-click the foreign table you want to check, and click **Generate DDL Statement** to check the SQL statement used to create the foreign table.

Delete a foreign table

On the left-side navigation submenu, click **PG management**. On the PG management tab, right-click the foreign table you want to delete and select **Delete Table**. In the Delete message, click **Ok** to delete the foreign table.

2.6.5. Data analytics

2.6.5.1. Overview

The Data Analytics module of HoloStudio is seamlessly integrated with DataWorks for node scheduling and provides all-in-one, stable, and efficient extract, transform, load (ETL) services. It can also synchronize MaxCompute table schemas and data and allows you to upload local files for data analytics.

The Data Analytics module consists of the following submodules:

1. Folder: stores data analytics nodes, helping you manage data analytics nodes of each database.
2. Interactive Analytics Development: is integrated with DataWorks to schedule ETL nodes.
3. One-click MaxCompute table structure synchronization: allows you to create multiple foreign tables sourced from MaxCompute at a time.
4. One-click MaxCompute data synchronization: provides a visualized user interface for you to synchronize MaxCompute data to Hologres.
5. One-click local file Upload: allows you to upload local files to Hologres.

Folder

Folders store data analytics nodes, helping you manage data analytics nodes of each database.

On the left-side navigation submenu, click **Data Analytics**. On the Data Analytics tab, move the pointer over the Create icon and select **Folder**. In the Create Folder dialog box, enter a folder name and click **Commit**.

2.6.5.2. Use the Interactive Analytics Development submodule

The Interactive Analytics Development submodule is seamlessly integrated with DataWorks. You can use HoloStudio to import data from MaxCompute to Hologres. You can also use DataWorks to schedule nodes to periodically import data to Hologres. This topic describes how to use HoloStudio to map the source data stored in a MaxCompute table to Hologres for periodic scheduling.

1. Prepare a MaxCompute table.

Create a table in MaxCompute and import data to the table. You can also select a table with data from Data Map. In this example, an existing table in Data Map is used. The following Data Definition Language (DDL) statement is used to create the table:

```
CREATE TABLE IF NOT EXISTS bank_data_odps
(
  age      BIGINT COMMENT 'age',
  job      STRING COMMENT 'job type',
  marital  STRING COMMENT 'marital status',
  education STRING COMMENT 'education level',
  card     STRING COMMENT 'credit card available or not',
  housing  STRING COMMENT 'mortgage',
  loan     STRING COMMENT 'loan',
  contact  STRING COMMENT 'contact',
  month    STRING COMMENT 'month',
  day_of_week STRING COMMENT 'day in a week',
  duration STRING COMMENT 'duration',
  campaign BIGINT COMMENT 'number of contacts during the campaign',
  pdays    DOUBLE COMMENT 'interval from the last contact',
  previous DOUBLE COMMENT 'number of contacts with the customer',
  poutcome STRING COMMENT 'result of the previous marketing campaign',
  emp_var_rate DOUBLE COMMENT 'employment change rate',
  cons_price_idx DOUBLE COMMENT 'consumer price index',
  cons_conf_idx DOUBLE COMMENT 'consumer confidence index',
  euribor3m DOUBLE COMMENT 'euro deposit rate',
  nr_employed DOUBLE COMMENT 'number of employees',
  y        BIGINT COMMENT 'fixed time deposit available or not'
);
```

2. Create a foreign table.

Go to the HoloStudio page. On the left-side navigation submenu, click **PG management** or **SQL Console**. On the tab that appears, create a foreign table for mapping data in the MaxCompute source table. In this example, use the following SQL statements to create a foreign table:

```
BEGIN;
CREATE FOREIGN TABLE if not EXISTS bank_data_foreign_holo (
  age int8,
  job text,
  marital text,
  education text,
  card text,
  housing text,
  loan text,
  contact text,
  month text,
  day_of_week text,
  duration text,
  campaign int8,
  pdays float8,
  previous float8,
  poutcome text,
  emp_var_rate float8,
  cons_price_idx float8,
  cons_conf_idx float8,
  euribor3m float8,
  nr_employed float8,
  y int8
)
SERVER odps_server
OPTIONS (project_name 'projectname', table_name 'bank_data_odps');
GRANT SELECT ON bank_data_foreign_holo TO PUBLIC;
COMMIT;
```

 **Note** The OPTIONS parameter contains two fields: project_name, which is the name of the MaxCompute project, and table_name, which is the name of the MaxCompute table.

3. Create a data storage table.

Create a table in HoloStudio to receive and store data. The fields in this table must be of the same data types as those in the foreign table. In this example, use the following SQL statements to create the storage table:

```
BEGIN;
CREATE TABLE if not EXISTS bank_data_holo (
  age int8,
  job text,
  marital text,
  education text,
  card text,
  housing text,
  loan text,
  contact text,
  month text,
  day_of_week text,
  duration text,
  campaign int8,
  pdays float8,
  previous float8,
  poutcome text,
  emp_var_rate float8,
  cons_price_idx float8,
  cons_conf_idx float8,
  euribor3m float8,
  nr_employed float8,
  y int8,
  ds text NOT NULL
)
PARTITION BY LIST(ds);
CALL SET_TABLE_PROPERTY('bank_data_holo', 'orientation', 'column');
CALL SET_TABLE_PROPERTY('bank_data_holo', 'time_to_live_in_seconds', '700000');
COMMIT;
```

4. Create a partitioned table.

On the HoloStudio page, click **Data Analytics** on the left-side navigation submenu. On the Data Analytics tab, move the pointer over the Create icon and select **Interactive Analytics Development** to create a Hologres development node. Then, go to the SQL editor of the node and enter SQL statements to create a partitioned table for obtaining the required data. After you enter the SQL statements, click the **Run** icon. In the Field dialog box, set a value for the `${bizdate}` parameter. After the SQL statements are executed, click the **Save** icon and then **Go to DataStudio for Scheduling** to schedule the node. You can enter the following sample SQL statements:

```
create table if not exists bank_data_holo_1_${bizdate} partition of bank_data_holo
for values in ('${bizdate}');
insert into bank_data_holo_1_${bizdate}
select
  age as age,
  job as job,
  marital as marital,
  education as education,
  card as card,
  housing as housing,
  loan as loan,
  contact as contact,
  month as month,
  day_of_week as day_of_week,
  duration as duration,
  campaign as campaign,
  pdays as pdays,
  previous as previous,
  poutcome as poutcome,
  emp_var_rate as emp_var_rate,
  cons_price_idx as cons_price_idx,
  cons_conf_idx as cons_conf_idx,
  euribor3m as euribor3m,
  nr_employed as nr_employed,
  y as y,
  '${bizdate}' as ds
from bank_data_foreign_holo;
```

5. Schedule the partitioned table.

Go to the DataStudio page and create a Hologres development node. In the SQL editor of the node, enter SQL statements to synchronize the partitioned table information to the node and click **Update Code**. Before you create the node, make sure that a workflow is created.

6. Set parameters for scheduling the data analytics node.

On the editing tab of the Hologres development node, click the **Properties** tab in the right-side navigation pane to set parameters for scheduling the node.

i. Set parameters in the General section.

In the **Arguments** field, specify a value for the `${bizdate}` variable.

ii. Set parameters in the Schedule section.

Select **Normal** for **Execution Mode** and set other parameters as required.

iii. Set parameters in the Dependencies section.

Select **Yes** for **Auto Parse** and click **Use Root Node**. After DataStudio automatically parses and displays the root node as a parent node, change the value of **Auto Parse** to **No**. You can also select a table that is scheduled as a parent node.

7. Save and deploy the node for scheduling.

After you set the scheduling parameters for the node, click the **Save** icon and then the **Submit** icon. After that, click **Deploy** in the upper-right corner.

8. Deploy the node in Operation Center.

On the **Create Package** page, find the target node and click **Publish** in the **Actions** column. After the node is deployed, click **Operation Center** in the top navigation bar to generate retroactive data for the node.

In **Operation Center**, right-click the published node and choose **Run > Current Node Retroactively**. Configure the node based on your business requirements.

9. Check the content of the table in HoloStudio.

After the retroactive data generation node is run, go back to HoloStudio. On the left-side navigation submenu, click **PG management**. On the PG management tab, click a database and choose **Mode > public > Table**. Double-click the partitioned table that is scheduled and click **Data Preview** to check whether the data is imported to the table.

2.6.5.3. Create multiple foreign tables at a time

Seamlessly integrated with MaxCompute at the underlying layer, Hologres allows you to create foreign tables to query MaxCompute data in an accelerated manner. You can create multiple foreign tables at a time by using the `IMPORT FOREIGN TABLE` statement. To free you from SQL operations, HoloStudio provides the following submodule for you to create foreign tables in a visualized manner: **One-click MaxCompute table structure synchronization**.

1. Create a schema sync node.

On the HoloStudio page, click **Data Analytics** on the left-side navigation submenu. On the Data Analytics tab, move the pointer over the **Create** icon and select **One-click MaxCompute table structure synchronization**. In the **Create Node** dialog box, set relevant parameters and click **Commit**. The schema sync node is created.

2. Set parameters for synchronizing the table schema.

After the schema sync node is created, you must set parameters for synchronizing the table schema based on your needs.

Parameter	Description	Remarks
Target Library	The name of the Hologres database where the foreign tables are to be created.	N/A
Target Schema	The name of the schema in the specified Hologres database.	The default value is public. If you have created a schema, you can select the created schema.
Remote Service type	The type of the external service. You can create only foreign tables sourced from MaxCompute.	The default value is odps.
Remote server	The external server. The default value is odps_server.	After you purchase a Hologres instance, the system automatically creates a server named odps_server. You can directly use it.
Remote library	The name of the MaxCompute project where the tables mapping the foreign tables to be created reside.	N/A

Parameter	Description	Remarks
Table name rules	The regular expression for specifying the tables whose schema is to be synchronized. By default, the schema of all tables in the specified MaxCompute project will be synchronized.	<ul style="list-style-type: none"> ◦ If a foreign table to be created is named the same as an existing foreign table in Hologres, the foreign table is not created. ◦ If a MaxCompute table whose schema is to be synchronized contains data types that Hologres does not support, an error is thrown. In this case, exclude this MaxCompute table in the regular expression. ◦ For more information, see <code>IMPORT FOREIGN SCHEMA</code>.
Regular preview	The execution result of the regular expression.	N/A

3. Run the schema sync node.

Click the **Save** icon and then click the **Run** icon to run the schema sync node. After the schema sync node is run, click PG management on the left-side navigation submenu. The created foreign tables appear. You can query the table data.

2.6.5.4. Import MaxCompute data

To improve the efficiency of querying MaxCompute data, Hologres allows you to import MaxCompute data to Hologres for queries. HoloStudio provides the following submodule for you to directly import MaxCompute data in a visualized manner: One-click MaxCompute data synchronization.

1. Create a data sync node.

On the HoloStudio page, click Data Analytics on the left-side navigation submenu. On the Data Analytics tab, move the pointer over the Create icon and select **One-click MaxCompute data synchronization**. In the Create Node dialog box, enter the node information and click Commit. The data sync node is created.

2. Set parameters for synchronizing data.

After the data sync node is created, you must set parameters for synchronizing data.

Section	Parameter	Description	Remarks
MaxCompute Source table selection	External table source	The source of the foreign table. Valid values: External table already exists and New external table.	<ul style="list-style-type: none"> ◦ If you select External table already exists, the existing foreign table mapping the MaxCompute table will be used. ◦ If you select New external table, you must create a foreign table mapping the MaxCompute table.

Section	Parameter	Description	Remarks
	External table name	The name of the existing foreign table.	The foreign table must map the MaxCompute table whose data will be synchronized.
Target table settings	Target Library	The name of the Hologres database to which the MaxCompute data will be synchronized.	N/A
	Target schema	The name of the schema in the specified Hologres database.	The default value is public. If you have created a schema, you can select the created schema.
	Destination Table Name	The name of the target table to which the MaxCompute data will be synchronized.	The table name can be customized.
	Target table description	The description of the target table.	N/A
Synchronization settings	Synchronization field	The fields to be synchronized from the specified MaxCompute table.	You can select specific or all fields in the MaxCompute table.
	Partition configuration	The partition fields to be synchronized.	Hologres supports a maximum of one level of partitions.
	Index configuration	The index to be built for the target table.	N/A
SQL Script	SQL Script	The SQL statements that are executed when the data sync node is run.	N/A

3. Run the data sync node.

Click the **Save** icon and then click the **Run** icon to run the data sync node. After the node is run, you can query the imported data in SQL Console or PG management.

2.6.5.5. Upload local files

This topic describes how to upload local files in HoloStudio in a visualized manner.

Hologres allows you to use the COPY statement to import data from the standard input of a client to a specified table. For more information, see COPY. HoloStudio allows you to import data in a local file to a specified table by uploading the local file in a visualized manner. To upload a local file in HoloStudio, perform the following steps:

1. Create a table.

In SQL Console or PG management, create a table to which data in the local file will be imported. In this example, use the following SQL statements to create a table:

```
BEGIN;
CREATE TABLE if not EXISTS holo_bank (
  age int8,
  job text,
  marital text,
  education text,
  card text,
  housing text,
  loan text,
  contact text,
  month text,
  day_of_week text,
  duration text,
  campaign int8,
  pdays float8,
  previous float8,
  poutcome text,
  emp_var_rate float8,
  cons_price_idx float8,
  cons_conf_idx float8,
  euribor3m float8,
  nr_employed float8,
  y int8
);
COMMIT;
```

2. Create a node for uploading the local file.

Go to the HoloStudio page. On the left-side navigation submenu, click Data Analytics. On the Data Analytics tab, move the pointer over the Create icon and select **Upload files locally with one click**.

3. Enter the node information.

In the **One-click local file Upload** dialog box, set the parameters based on your business needs and click Next Step.

Parameter	Description	Remarks
Target Library	The name of the Hologres database where the target table resides.	N/A
Target Schema	The name of the schema where the target table resides.	The default value is public. If you have created a schema, you can select the created schema.
Select the data table to import	The name of the target table to which data in the local file will be imported.	N/A

4. Select the local file to upload and set other required parameters.

After you click Next Step, select the local file to upload, set other required parameters, and then click Commit.

Parameter	Description	Remarks
-----------	-------------	---------

Select File	The local file to upload.	You can select a local file only in the .txt, .csv, or .log format.
Select separator	The delimiter of fields in the file. Select Comma (,) or Space ().	N/A
Original character set	The character set of the file.	<ul style="list-style-type: none"> ◦ GBK ◦ UTF-8 ◦ CP936 ◦ ISO-8859
First behavior title	Specifies whether to use the first line as the header line.	N/A

5. View the imported data.

After you click **Commit**, data in the selected local file is imported to the specified table. You can go to SQL Console or PG management to view the imported data.

2.6.6. Hologres console

2.6.6.1. Overview

Hologres is a real-time interactive analytics service that is fully compatible with PostgreSQL and seamlessly integrated with the big data ecosystem. Hologres delivers high-concurrency and low-latency performance in analyzing terabyte-scale data. Hologres allows you to use mainstream Business Intelligence (BI) tools to get an analytical insight into data from multiple dimensions and explore business data in an efficient and cost-effective manner.

For convenience of business, Apsara Stack provides the Hologres console independent from the DataWorks console for different users to managing Hologres instances, users, and databases.

Log on to the Apsara Stack console. In the top navigation bar, choose **Products > Interactive Analytics** to go to the Hologres console. The following figure shows the Overview page in the Hologres console.

2.6.6.2. View the instance list

The Instances page lists all Hologres instances purchased by your Apsara Stack tenant account. On this page, you can view the instance status, change instance configurations, and create instances. You can also click an instance name to go to the instance details page where you can manage objects in the instance, including databases and users.

Instances

Log on to the Apsara Stack console. In the top navigation bar, choose **Products > Interactive Analytics** to go to the Hologres console. In the Hologres console, click **Instance List** in the left-side navigation pane. The following figure shows the Instance List page.

1. New engine instance button

On the Instance List page, click **New engine instance**. In the New engine instance dialog box, enter an instance name and select the instance specifications to create a Hologres instance.

2. Search box

If you have purchased multiple Hologres instances, you can enter a keyword of an instance name in the search box to find the target instance.

3. Running status column

The Running status column displays the running status of each Hologres instance. An instance can be in one of the following states:

- Normal operation: The instance is running as expected.
- Creating: The payment is successful, and Hologres is creating the instance. You must wait for 3 to 5 minutes.
- Shutdown: The instance has been suspended and you cannot connect to it.

Operation column

The Operation column provides the following buttons for you to manage a Hologres instance:

1. Management

Find the target instance and click **Management** in the Operation column. On the page that appears, you can view and manage objects in the instance, including databases and users.

2. Change configuration

If your instance cannot meet your business needs or your instance has a large amount of surplus resources, you can click **Change configuration** in the Operation column. In the **Change configuration** dialog box, upgrade or downgrade your instance configurations based on your business needs.

3. Shutdown

Find the target instance and click **Shutdown** in the Operation column to suspend the instance. You cannot connect to the suspended instance.

2.6.6.3. Manage instances

This topic describes how to view and change instance configurations, select a network type, and select a connection method on the Basic information page in the Hologres console.

View and change instance configurations

In the Hologres console, click **Instance List** in the left-side navigation pane. In the instance list, find the target instance and click **Management** in the Operation column. The Basic information page displays basic information about a Hologres instance, including the instance name, instance ID, region, instance version, billing method, instance specification, and creation time.

If you need to change the instance specifications, click **Change configuration**. In the Change configuration dialog box, upgrade or downgrade the instance specifications based on your business needs.

Select a network type

The following table lists the supported network type.

Network type	Domain name	Scenario
Internal network	<instancename>-cn-<region>-internal.hologres.aliyuncs.com:80	Select this network type when you want to connect to the Hologres instance by using the classic network, without charges on the Internet traffic.

Select a connection method

Hologres is compatible with PostgreSQL. You can connect to a Hologres instance from the PostgreSQL client or over JDBC interfaces by using ETL or BI tools.

The Connection Methods section offers methods for you to use common development tools to connect to a Hologres instance. You can select a development tool and connection method based on your business needs and preference.

1. Connect from the PostgreSQL client

To connect to a Hologres instance from the PostgreSQL client, use the following connection string:

```
PGUSER=<AccessId> PGPASSWORD=<AccessKey> psql -p <Port> -h <Endpoint> -d <Database>
```

2. Connect over JDBC

To connect to a Hologres instance over JDBC, use the following connection string:

```
postgres://<AccessId>:<AccessKey>@<Endpoint>:<Port>/<database>?preferQueryMode=simple&tcpKeepAlive=true
```

2.6.6.4. Manage users

This topic describes how to manage users on the User Management page in the Hologres console.

Overview

In the Hologres console, click **Instance List** in the left-side navigation pane. In the instance list, find the target instance and click **Management** in the **Operation** column. On the page that appears, click **User Management**. On the User Management page, you can manage users on a Hologres instance without executing cumbersome SQL statements. For example, you can add and delete users and grant permissions to users on this page.

After you create a Hologres instance with your Apsara Stack tenant account, this account becomes a superuser of the instance. A superuser has all permissions on the Hologres instance. By default, the User Management page displays only the information of the Apsara Stack tenant account that creates the Hologres instance. The information of a Resource Access Management (RAM) user appears on this page only after you use the Apsara Stack tenant account to add it to the instance.

Column	Description	Remarks
Members	Displays the usernames of the Apsara Stack tenant account and RAM users on the Hologres instance.	Generally, a username appears in the xxx format.
Cloud account	Displays the account IDs of users on the Hologres instance.	N/A
Type	Displays the roles assigned to users on the Hologres instance.	The user can be a superuser or normal user.

Add a user

On the User Management page, you can create RAM users on a Hologres instance without executing the SQL CREATE statement.

Click **Add new user**. In the Add new user dialog box, select existing RAM users under your Apsara Stack tenant account to add them to the Hologres instance. If no RAM user exists under your Apsara Stack tenant account, create a RAM user first.

When you add a RAM user, you can assign the superuser or normal user role to the user.

- **Superuser:** A superuser has all permissions on the Hologres instance without the need for additional authorization.
- **Normal user:** A normal user cannot view or manage any objects on the Hologres instance, including databases, schemas, and tables. A normal user must be authorized before it can view and manage objects in the instance. We recommend that you go to the DB management page to grant permissions to RAM users as required. Alternatively, you can use SQL statements to grant permissions to RAM users.

Delete a user

Find the target user on the User Management page and click **Delete** in the Operation column to delete the user from the Hologres instance. A deleted user has no access to the Hologres instance.

2.6.6.5. Manage databases

This topic describes how to manage databases on the DB management page in the Hologres console.

Overview

In the Hologres console, click **Instance List** in the left-side navigation pane. In the instance list, find the target instance and click **Management** in the **Operation** column. On the page that appears, click **DB management**. On the DB management page, you can manage all databases on the current Hologres instance. You can create databases, select a permission management mode for the databases, and view database information.

 **Note** A default database named `postgres` is automatically created after you create a Hologres instance. This database is provided for management purposes only and does not appear on the DB management page. This database is allocated with limited resources. Create databases on this page based on your business needs.

Create a database

Hologres allows you to create a database with one click on the graphical user interface (GUI), eliminating the need for SQL operations.

Click **New Database**. In the New Database dialog box, enter a name for the database and set the Simple permissions model parameter to Open or Close. To simplify authorization, we recommend that you set the Simple permissions model parameter to Open.

Hologres provides two permission models for you to authorize users in a convenient way.

- **Standard PostgreSQL authorization:** Compatible with PostgreSQL, Hologres provides a permission model that is exactly the same as the standard PostgreSQL authorization model. You can authorize RAM users by using the standard PostgreSQL GRANT statement.
- **SPM:** Backed by the understanding of customers' business and its practical experience, Alibaba Cloud introduced a simple permission model (SPM) to Hologres to simplify the management of user permissions. The SPM is a coarse-grained model that authorizes users by user group.

After a database is created, you can use a development tool to connect to the database to analyze data.

Authorize a user

After the SPM is enabled for a new database, you can authorize RAM users with one click in the Hologres console. Perform the following steps:

1. **Open the Permission management right-side pane.**

Find the target RAM user and click **User authorization** in the Operation column. You can grant permissions to a RAM user by adding the user to the desired user group.

2. Add a RAM user to a user group.

In the Permission management right-side pane, click **Add authorization**. In the Add authorization dialog box, select the account to which you want to grant permissions, select the desired user group below Permissions policy, and then click **OK**.

Revoke permissions

If the SPM is enabled for your database, you can revoke the permissions of a RAM user with one click in the Hologres console.

On the instance details page, click **DB management**. On the DB management page, find the target database and click **User authorization** in the Operation column. In the Permission management right-side pane, find the target RAM user and click **Delete authorization** in the Operation column.

Delete a database

On the DB management page, find the database no longer required and click **Delete** in the Operation column to delete the database. After a database is deleted, data in the database is also deleted and cannot be recovered.

2.7. Realtime Analysis

2.7.1. Overview

Integrated with DataWorks, DataAnalysis supports creating MaxCompute tables in tabular mode, collaboratively editing workbooks and performing statistical analysis, and generating and sharing visual reports. These features enable data developers and business staff to quickly analyze data.

Go to DataAnalysis

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > DataAnalysis**.
3. On the **Home** page of DataAnalysis, click **Experience now** to go to the Web Excel page.

Features

- **Workbook**

You can create and edit workbooks. Workbooks support basic operations such as addition, subtraction, multiplication, and division, and multiple data processing methods, including functions, classification, and aggregation. In addition, you can edit workbooks collaboratively with other users online and create pivot tables for further analysis.

- **Visual report**

You can create and design visual reports by dragging, dropping, and configuring controls without running SQL statements.

- **Dimension table**

You can create MaxCompute tables in tabular mode by one click without running SQL statements and edit MaxCompute tables collaboratively with other users online.

- You can create a MaxCompute table in tabular mode.
- You can import data into a MaxCompute table by one click.

2.7.2. Workbook

2.7.2.1. Create a workbook

This topic describes how to create a workbook. After a workbook is created, you can rename, clone, delete, or change the owner of the workbook.

Procedure

1. Log on to the DataWorks console.
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**.
4. On the **Web Excel** page, click  in the **New Spreadsheet** section.

 **Note** If you have created workbooks, you can search for a workbook by entering its name in the search box in the **All Spreadsheets** section. Then, click the workbook name in the **File Name** column to go to the workbook editing page.

5. In the **New spreadsheet** dialog box, enter a name in the **File Name** field.
6. Click **OK**.

Result

After the workbook is created, it appears in the **All Spreadsheets** section. In this section, you can view all created workbooks. In addition, you can rename, clone, delete, or change the owner of a workbook.

- Find the target workbook and click **Rename** in the Operation column. In the **Rename** dialog box, enter the new name in the **File Name** field and click **OK**.
- Find the target workbook and click **Change Owner** in the Operation column. In the **Change Owner** dialog box, select an owner from the New Owner drop-down list and click **OK**.
- Find the target workbook and click **Clone** in the Operation column. The cloned workbook appears in the workbook list. The name of the cloned workbook contains the **_copy** suffix.
- Find the target workbook and click **Delete** in the Operation column. In the **Delete** message, click **OK**.

2.7.2.2. Edit a workbook

This topic describes how to edit a workbook. For example, you can import data to, export data from, and share a workbook, create a pivot table in a workbook, and use the data profiling feature.

Go to the workbook editing page

1. [Log on to the DataWorks console](#).
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**. The **Web Excel** page appears.
4. In the **All Spreadsheets** section of the **Web Excel** page, click the name of the target workbook in the **File Name** column. After you create a workbook, the workbook editing page appears.

Apply a template to a workbook or save a workbook as a template

You can apply an existing template to the current workbook by performing the following steps:

1. In the upper-right corner of the workbook editing page, choose **Template > Import Template**.
2. In the **Import Template** dialog box, select a file to be used as a template for the current workbook.

Note The data of the selected template will overwrite that of the current workbook.

3. Click **OK**.

You can save the current workbook as a template by performing the following steps:

1. In the upper-right corner of the workbook editing page, choose **Template > Save as Template**.
2. In the **Template settings** dialog box, set the **Type**, **Name**, and **Description** parameters.

Notice The template name can be up to 256 characters in length and the template description can be up to 1,024 characters in length.

3. Click **OK**.

Import data to a workbook

1. In the upper-right corner of the workbook editing page, click **Import**.
2. In the Open dialog box, find and select a local file to be imported and click **Open**. The data in the local file is imported to the current workbook.

Note You can only import data from Excel files.

Export data from a workbook to a MaxCompute table

1. In the upper-right corner of the workbook editing page, choose **Export > Generate MaxCompute Build Table Statement**.
2. In the **Export as MaxCompute Table** dialog box, set relevant parameters.

Insert mode	Parameter	Description
Insert Data into MaxCompute Table (INSERT OVERWRITE)	Workspace	The workspace to which the MaxCompute table belongs.

TABLE (INSERT OVERWRITE) (insertoverwrite)	Parameter	Description
	Table	The MaxCompute table to which you want to insert data.
Create MaxCompute Table and Insert Data (INSERT OVERWRITE)	Workspace	The workspace to which the MaxCompute table belongs.
	Table Name	The name of the MaxCompute table. Make sure that the table name has not been used. You can click Check Duplicate Names to check whether the table name exists.

3. After the parameters are set, click **Copy SQL Statement**.

 **Notice** Only non-partitioned tables are supported.

Create a pivot table

1. On the workbook editing page, select the data for which you want to create a pivot table and click **Pivot** in the upper-right corner.
2. In the **Create Pivot Table** dialog box, set relevant parameters.

Specify the range of the data to be analyzed. You can set the **Choose Data** parameter to **Select Range** or **Use External Data Store** as needed.

- If you select **Select Range**, select the cells in the workbook for which you want to create a pivot table. The value of the **Range** field changes based on the selected cells.

 **Note** This parameter is available only when you select **Select Range**.

- If you select **Use External Data Store**, set the **Type** parameter first. You can set the **Type** parameter to **Mysql** or **Data Services**.

- If you select **MySQL**, set the parameters described in the following table.

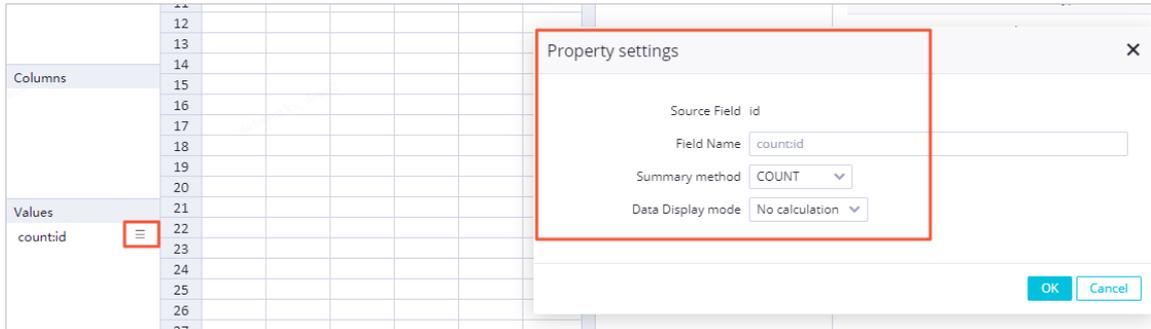
Parameter	Description
Choose Data	The range of the data to be analyzed. Select Use External Data Store .
Type	The type of the data source. Select MySQL .
Workspace	The workspace where the MySQL data store resides.
Data Store	The name of the connection to the data store. To create a connection to a data store, perform the following steps: Click the Workspace Manage icon in the upper-right corner. On the page that appears, click Data Source in the left-side navigation pane. On the Data Source page, create a connection to a data store.
Table	The table for which you want to create a pivot table.

- If you select **Data Services**, set the parameters described in the following table.

Parameter	Description
Choose Data	The range of the data to be analyzed. Select Use External Data Store .
Type	The type of the data source. Select Data Services .
Workspace	The workspace where the API of DataService Studio resides.
API Group	The API group to which the API belongs.
API	The API to be used as the data source.

3. Click **OK**. The pivot table editing page appears. This topic takes the pivot table for a selected range of data as an example.

- **Data Source**: the range that you specified in the previous step.
- **Pivot Table Fields**: the names of the fields that you selected in the previous step.
- **Rows**: Drag fields from the Pivot Table Fields section to the **Rows** section. Each value of the field added to the **Rows** section occupies a row in the pivot table.
- **Columns**: Drag fields from the Pivot Table Fields section to the **Columns** section. Each value of the field added to the **Columns** section occupies a column in the pivot table.
- **Values**: Click the property setting icon for a field in the **Values** section. In the Property settings dialog box, set the **Summary method** and **Data Display mode** parameters. By default, the **Field Name** parameter cannot be modified.



Parameter	Description
Source Field	The name of the selected source field.
Field Name	The name of the field that appears in the pivot table. The name is in the format of Aggregation method:Source field name .
Summary method	The aggregation method. Valid values: SUM, COUNT, MAX, MIN, and AVG .
Data Display mode	The mode for displaying the data. Valid values: No calculation and Percentage of Total .

- **Filters:** Drag fields from the Pivot Table Fields section to the Filters section. In the right-side pivot table display area, you can select the fields to filter data.

Download a workbook

In the upper-right corner of the workbook editing page, click **Download** to download the workbook to a local directory.

Share a workbook

In the upper-right corner of the workbook editing page, click **Share**. In the dialog box that appears, set the sharing mode.

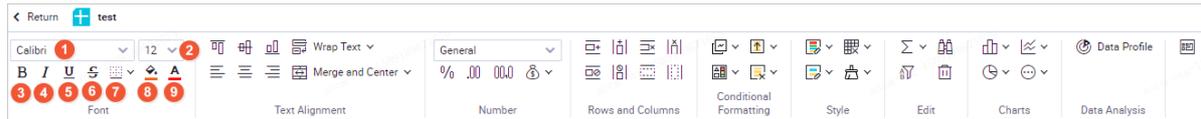
You can share the workbook in the following ways:

- **Link:** Click **Copy Link** and send the copied URL to other users as needed.
- **Users with Edit Access:** Click **Add** in the **Users with Edit Access** section. In the dialog box that appears, select the users to whom you want to grant the edit permission and click **OK**.
- **Visible to All:** To allow all users to view the workbook, turn on the **Visible to All** switch.
- **Users with Read Access:** To allow only specific users to view the workbook, turn off the **Visible to All** switch and click **Add** in the **Users with Read Access** section. In the dialog box that appears, select the users to whom you want to grant the read-only permission and click **OK**.

Note If the system notifies you that the number of users to whom you want to grant the read-only permission reaches the upper limit, you can upgrade the DataWorks edition.

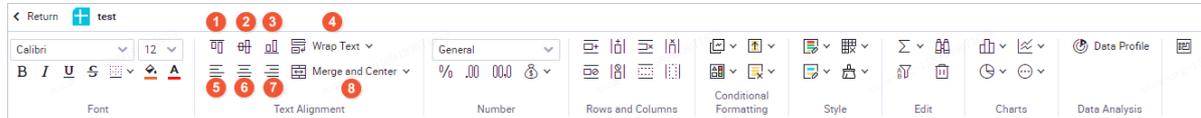
Menu bar

- **Font**



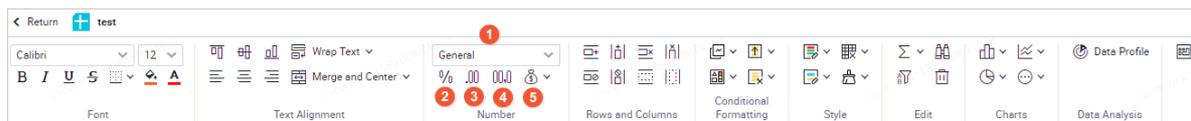
No.	Feature	Description
①	Font	Select a font type from the drop-down list as needed.
②	Font Size	Select a font size from the drop-down list as needed.
③	Bold	Set text in bold.
④	Italic	Set text in italic.
⑤	Underline	Underline text.
⑥	Strikethrough	Add a strikethrough to text.
⑦	Borders	Add borders to cells.
⑧	Background Color	Specify the background color of cells.
⑨	Text Color	Change the text color.

● **Text Alignment**



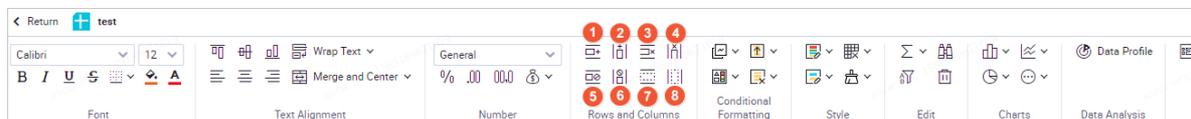
No.	Feature	Description
①	Top Align	Align text vertically to the top.
②	Middle Align	Align text vertically to the center.
③	Bottom Align	Align text vertically to the bottom.
④	Wrap Text	Display long text in multiple lines in a cell.
⑤	Align Left	Align text horizontally to the left.
⑥	Center	Align text horizontally to the center.
⑦	Align Right	Align text horizontally to the right.
⑧	Merge and Center	Merge multiple cells to one cell and center the content in the cell.

● **Number**



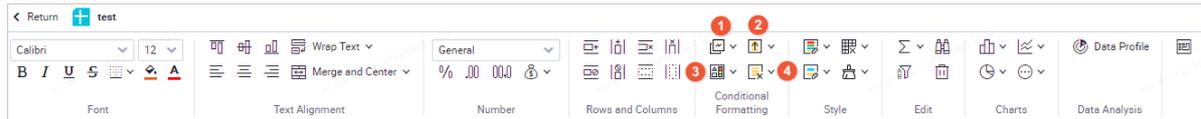
No.	Feature	Description
①	Data Type	Specify the type of data held in cells. You can select General, Number, Currency, Short Date, Long Date, Time, Percentage, Fraction, Scientific, and Text from the drop-down list.
②	Percentage	Apply the percentage format to numbers.
③	Two Decimal Places	Round numbers to two decimal places.
④	1000 Separator	Display numbers with thousands separators, for example, 1,005.
⑤	Currency	Add a currency sign to numbers. The following currency signs are supported: yuan sign (¥), dollar sign (\$), pound sign (£), euro sign (€), and franc sign (Fr).

• **Rows and Columns**



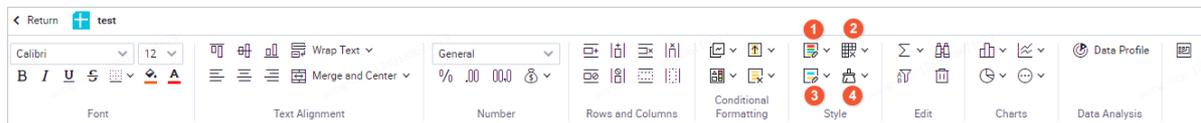
No.	Feature	Description
①	Insert Row	Insert a row to the workbook.
②	Insert Column	Insert a column to the workbook.
③	Delete Row	Delete rows from the workbook.
④	Delete Column	Delete columns from the workbook.
⑤	Lock Row	Lock the rows before the selected row in the workbook.
⑥	Lock Column	Lock the columns before the selected column in the workbook.
⑦	Hide Row	Hide rows in the workbook.
⑧	Hide Column	Hide columns in the workbook.

• **Conditional Formatting**



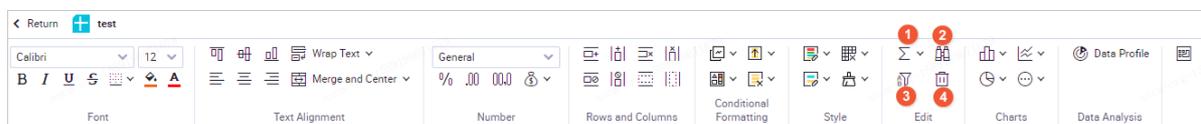
No.	Feature	Description
①	Highlight cell rules	Specify the rules for highlighting cells.
②	Data Bar/Color Scale	Format cells by using data bars and color scales.
③	Icon Set	Format cells by using icon sets. The icon sets include directional icons, shapes, indicators, and rating icons.
④	Clear Rule	Clear the formatting. You can select Clear Rules from Selected Cells or Clear Rules from Entire Sheet from the drop-down list.

• **Style**



No.	Feature	Description
①	Apply table style	Apply a predefined table style to cells.
②	Delete	Remove the applied table style.
③	Cell Style	Apply a cell style to cells.
④	Clear	Clear the content or style in cells. You can select Clear All , Clear Content , or Clear Style from the drop-down list.

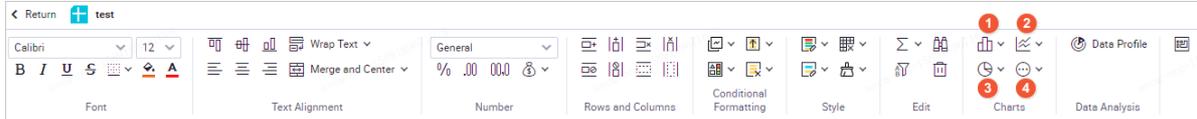
• **Edit**



No.	Feature	Description
①	AutoSum	Select an aggregation method. You can select Sum , Average , Count Numbers , Max , or Min from the drop-down list.
②	Search	Click Search or press Ctrl+F to open the search box.
③	Sort and Filter	Filter data and sort data in ascending or descending order.

No.	Feature	Description
④	Clear	Clear the content in cells.

• **Charts**



- **Column Chart** : After you click the **Column Chart** icon, you can select **Column Chart** , **Stacked Column Chart** , or **100% Stacked Column Chart** .
- **Line Chart** : After you click the **Line Chart** icon, you can select **Line Chart** , **Stacked Line Chart** , **100% Stacked Line Chart** , **Line Chart with Markers** , **Stacked Line Chart with Markers** , or **100% Stacked Line Chart with Markers** .
- **Pie Chart** : After you click the **Pie Chart** icon, you can select **Pie Chart** or **Doughnut Chart** .
- **More** : After you click the **More** icon, you can view more chart types, including area charts, bar charts, scatter charts, and stock charts.

• **Data Profile**

The data profiling feature allows you to analyze the quality, structure, distribution, and statistics of the data. It also allows you to preview, profile, process, analyze, and visualize data. The data profiling feature analyzes data by column and allows you to view the distribution of data types and values of each column.

Select the data to be analyzed and click **Data Profile**. The data profiling feature displays the data type and value distribution of each column above the editing area in the form of charts and rich text.

	A	B	C	D	E	F
unique values	13			12		
string		77%	100%		100%	
bigint		23%				

Simple mode:

- For a column whose values are of the **STRING** or **DATE** type: The simple mode displays the values ranking top 2 based on frequency and their respective percentages, and the percentage of other values in the form of rich text. If the number of value types exceeds 50% of the total number of values, the simple mode displays the number of unique values.
- For a column whose values are of the **INTEGER** or **FLOAT** type: The simple mode displays the value distribution in the form of a histogram.
- For a column whose values are of the **BOOLEAN** type: The simple mode displays the proportions of different values in the form of pie charts.
- For a column whose values are of two or more data types: The simple mode displays the proportions of different data types in the form of pie charts. In addition, the system reminds you that the current column has dirty data. After the dirty data is cleared, the simple mode displays the data in one of the preceding forms based on the data type.
- For a column whose values are null values: The simple mode displays the percentage of null values in red.

Click **Detailed Mode** in the upper-right corner. In the **Data Profile** dialog box, you can view the profiling result of each column.

Detailed mode:

- For a column whose values are of the STRING or DATE type: The detailed mode displays the number of fields, the numbers and percentages of unique values, valid values, and null values, and the numbers of occurrences of the values ranking top 5 based on frequency.
- For a column whose values are of the INTEGER or FLOAT type: The detailed mode displays the number of fields, the numbers and percentages of unique values, valid values, zeros, and null values, the numbers of occurrences of the values ranking top 5 based on frequency, the statistics, and a histogram.
- For a column whose values are of the BOOLEAN type: The detailed mode displays the number of fields, the numbers and percentages of unique values, zeros, and null values, the numbers of occurrences of the values ranking top 5 based on frequency, and a pie chart.

 **Note** The system considers the true and false strings and the 0 and 1 integers as values of the BOOLEAN type.

- **List of Shortcut Keys**

Click  to view the shortcut keys for different features.

2.7.3. Dimension tables

2.7.3.1. Create and manage dimension tables

The dimension table feature allows you to create MaxCompute tables, import local data to MaxCompute tables, and edit MaxCompute tables in a visualized manner.

Prerequisites

1. A MaxCompute compute engine instance is bound to a DataWorks workspace.
2. A MaxCompute table is created. For more information about how to create a MaxCompute table, see [Create a MaxCompute table](#).

Limits

- To create a dimension table in DataAnalysis, you must be an administrator, a project owner, or a developer of a DataWorks workspace.
- For a MaxCompute table that is created by using the dimension table feature, all fields in the MaxCompute table are of the STRING type. If you want to use fields of other data types, execute data definition language (DDL) statements to create a MaxCompute table on the **DataStudio** page.

Create a dimension table

1. [Log on to the DataWorks console](#).
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Development > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**. The **Web Excel** page appears.
4. In the top navigation bar, click **Dimension**.
5. On the **Dimension** page, click the  icon in the **New Dimension Table** section.
6. In the **New Dimension Table** dialog box, configure the parameters as required.

Parameter	Description
-----------	-------------

Parameter	Description
Target Workspace	The DataWorks workspace to which the MaxCompute table belongs.
Table Name	<p>The name of the dimension table. A MaxCompute table will be used in the production environment.</p> <p> Note The table name can contain only letters, digits, and underscores (_), and must start with a letter.</p>
Table Description	The description of the table, such as the purpose or features.
Field	The fields in the table. Only fields of the STRING type can be added.
Lifecycle	The lifecycle of the table. The table occupies storage resources in MaxCompute. To make sure that the resources can be recycled, select a proper lifecycle for the table from the drop-down list. If the specified lifecycle is exceeded, the table is deleted.

7. Select **I have known this risk and confirmed that as owner of this table, I am responsible for the subsequent changes to this table.** and click **OK** to go to the dimension table editing page to view and modify information about the table. For more information about how to edit a dimension table, see [Edit a dimension table](#). The MaxCompute table created in DataAnalysis is maintained in the production environment. The creator of the table is responsible for the creation and maintenance of the table.

View and manage dimension tables

1. On the dimension table editing page, click **Return** in the upper-left corner or **Dimension** in the top navigation bar to go back to the Dimension page.
2. In the **All Dimension Tables** section, select **I created** or **Share it with me** from the drop-down list in the upper-right corner to view the tables in the corresponding category. You can also share dimension tables with specific members. For more information about how to share a dimension table, see [Share a dimension table](#).
3. Click the file name of a required table or click **Edit** in the Operation column of the table to go to the dimension table editing page. On the Dimension page, you can also perform the following operations to manage a dimension table:
 - o To change the owner of a dimension table, find the table and click **Change Owner** in the Operation column. In the **Change Owner** dialog box, select an owner from the New Owner drop-down list and click **OK**.
 - o To delete a dimension table, find the table and click **Delete** in the Operation column. In the **Delete** message, click **OK**.

What's next

After you create a dimension table, go to the dimension table editing page and import data to this table. For more information about how to import data to a dimension table, see [Import data to a dimension table](#).

2.7.3.2. Import data to a dimension table

After you create a dimension table, you can write data to the table for data analysis. You can also import data from a workbook, local CSV file, or local Excel file to the table for data analysis. This topic describes how to import data to a dimension table.

Prerequisites

A dimension table is created. For more information about how to create a dimension table, see [Create a dimension table](#).

Procedure

1. [Log on to the DataWorks console](#).
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Development > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**. The **Web Excel** page appears.
4. In the top navigation bar, click **Dimension**.
5. In the **All Dimension Tables** section of the **Dimension** page, click the name of the table that you want to edit in the **File Name** column to go to the dimension table editing page. If no dimension table exists and you created one, the table editing page appears after the table is created. For more information about how to edit a dimension table, see [Edit a dimension table](#).
6. On the dimension table editing page, click **Import** in the upper-right corner.
7. In the **Import** dialog box, select the type of the file that contains the data to be imported and configure parameters.

 **Note** Only data of the STRING type can be imported to a dimension table. Data that is not of the STRING type will be automatically converted to the STRING type when it is imported.

o Workbook

Parameter	Description
Spreadsheet	The workbook from which the data is to be imported. Select a workbook from the Spreadsheet drop-down list.
Sheet	The sheet from which the data is to be imported. Select a sheet from the Sheet drop-down list.
Data Preview	Displays the data in the selected sheet. When you preview the data in the selected sheet, you can determine whether to use the values in the first row as the column names by selecting or clearing First Row as Field Names .
Field Mapping	The mappings between the columns in the selected sheet and the fields in the dimension table.
Import Data Mode	The mode used to import data. Valid values: Append and Overlay .

o Local CSV file

Parameter	Description
File	The CSV file from which the data is to be imported. Click Select File(.csv) , select a CSV file from the on-premises machine, and then click Open .
Original Character Set	The character set that is used by the selected CSV file. Valid values: UTF-8 and GBK . If garbled characters appear, you can change the character set.

Parameter	Description
Separator	<p>The row delimiter and column delimiter.</p> <ul style="list-style-type: none"> Valid values of row delimiters: \r\n, \n, and \r. Valid values of column delimiters: commas (,), semicolons (;), and \t. <p>If the cell data cannot be correctly separated, you can change the delimiters.</p>
Data Preview	<p>Displays the data in the selected sheet. When you preview the data in the selected sheet, you can determine whether to use the values in the first row as the column names by selecting or clearing First Row as Field Names.</p>
Field Mapping	<p>The mappings between the columns in the selected sheet and the fields in the dimension table.</p>
Import Data Mode	<p>The mode used to import data. Valid values: Append and Overlay.</p>

o **Local Excel file**

Parameter	Description
File	<p>The Excel file from which the data is to be imported. Click Select File(.xlsx), select an Excel file from the on-premises machine, and then click Open.</p>
Sheet	<p>The sheet from which the data is to be imported. Select a sheet from the Sheet drop-down list.</p>
Data Preview	<p>Displays the data in the selected sheet. When you preview the data in the selected sheet, you can determine whether to use the values in the first row as the column names by selecting or clearing First Row as Field Names.</p>
Field Mapping	<p>The mappings between the columns in the selected sheet and the fields in the dimension table.</p>
Import Data Mode	<p>The mode used to import data. Valid values: Append and Overlay.</p>

- Click **OK**.
- Click **Save** in the upper-right corner of the page. After you save the dimension table, you can click **Diff** in the upper-right corner of the page to check whether the changes meet expectations to prevent misoperations.

2.7.3.3. Edit a dimension table

This topic describes how to edit a dimension table in a visualized manner to modify the information of a MaxCompute table that you created. You do not need to write SQL code.

Prerequisites

A dimension table is created. For more information, see [Create a dimension table](#).

Procedure

1. Log on to the DataWorks console.
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Development > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**. The **Web Excel** page appears.
4. In the top navigation bar, click **Dimension**.
5. In the **All Dimension Tables** section of the **Dimension** page, click the name of the table that you want to edit in the **File Name** column to go to the dimension table editing page.

6. On the dimension table editing page, view and modify the information about the dimension table. In the left-side pane of the dimension table editing page, you can view the table information, such as the workspace, table name, table description, lifecycle, and field description. To view the details of the dimension table, click the link below **Table Details** to go to the **Data Map** page. For more information, see [View the details of a table](#).

To modify the settings of the dimension table, perform the following steps: Click **Modify field settings**. In the **Modify the field settings dimension table** dialog box, change the values of **Table Description** and **Lifecycle** and click OK. You can also add fields to the table in this dialog box.

The right side of the dimension table editing page displays all the data in the MaxCompute table as a workbook. The values in the first row are field names. You can double-click a cell to modify the content of a field in the corresponding row.

7. Click **Save** in the upper-right corner of the page to save the changes. After you save the dimension table, you can view all the data in the table. You can also click **Diff** in the upper-right corner of the page to view all the data in the **Diff Results** dialog box.

2.7.3.4. Share a dimension table

If you want to collaboratively edit a dimension table with multiple members, you can share the dimension table and grant the members the permissions to edit the table. This topic describes how to share a dimension table and grant edit or read permissions to specified members.

Prerequisites

The **Allow Sharing** switch is turned on on the **Management** page.

Procedure

1. Log on to the DataWorks console.
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data Development > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**. The **Web Excel** page appears.
4. In the top navigation bar, click **Dimension**.
5. In the **All Dimension Tables** section of the **Dimension** page, click the name of the table that you want to edit in the **File Name** column to go to the dimension table editing page. If no dimension table exists and you created one, the table editing page appears after the table is created. For more information about how to edit a dimension table, see [Edit a dimension table](#).
6. In the upper-right corner of the dimension table editing page, click **Share**. In the dialog box that appears, configure the sharing method as required. You must configure the following information before you can share a dimension table with other members:
 - o **Link**: After you specify Users with Edit Access and Users with Read Access, click **Copy Link** and send the copied URL to specified members.

- **Users with Edit Access:** To specify members with permissions to edit the dimension table, click **Add** in the **Users with Edit Access** section. In the **Share File with These Users** dialog box, enter and select the names of the members to be granted the edit permissions, and click **OK**.

 **Note** You can grant the edit permissions to a maximum of 10 members.

- **Users with Read Access:** To specify members with permissions to read the dimension table, click **Add** in the **Users with Read Access** section. In the **Share File with These Users** dialog box, enter and select the names of the members to be granted the read permissions, and click **OK**.

 **Note** You can grant the read permissions to a maximum of 30 members.

After the sharing method is configured, you can send the URL to specified members. The members can use the URL to access the dimension table. You can go back to the **Dimension** page to view shared dimension tables.

2.7.4. Report

2.7.4.1. Create a report

This topic describes how to create a report. After a report is created, you can rename or delete the report.

Procedure

1. Log on to the DataWorks console.
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. On the DataAnalysis homepage, click **Experience Now**.
4. On the **Web Excel** page, click **Report** in the top navigation bar.
5. Click  in the **New Report** section. Alternatively, click a template in the **New Report** section to create a report.

 **Note** If you have created reports, you can search for a report by entering its name in the search box in the **All Reports** section. Then, click the report name in the **File Name** column to go to the report editing page.

6. In the **New Report** dialog box, set the **Report Name** and **Report Description** parameters.
7. Click **OK**.

Result

After the report is created, it appears in the **All Reports** section. In this section, you can view all created reports. In addition, you can rename or delete a report.

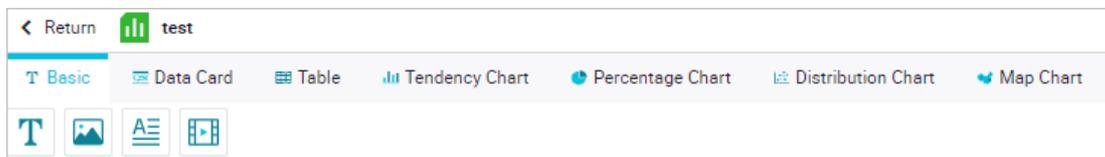
- To rename a report, perform the following steps: Find the target report and click **Rename** in the **Operation** column. In the **Rename** dialog box, enter the new name in the **File Name** field and click **OK**.
- To delete a report, perform the following steps: Find the target report and click **Delete** in the **Operation** column. In the **Delete** message, click **OK**.

2.7.4.2. Edit a report

This topic describes how to edit, preview, save, share, and release a report, and save a report as a template.

Procedure

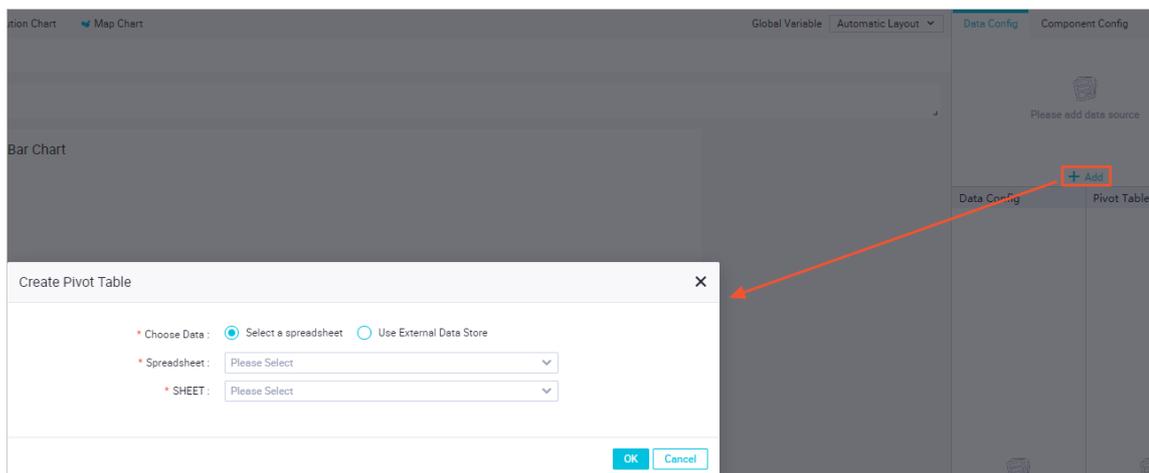
1. Go to the Report page.
 - i. Log on to the DataWorks console.
 - ii. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
 - iii. On the DataAnalysis homepage, click **Experience Now**.
 - iv. On the **Web Excel** page, click **Report** in the top navigation bar.
2. Go to the report editing page. Use one of the following methods to go to the report editing page:
 - o After you create a report, the report editing page appears.
 - o In the **All Reports** section of the Report page, click the name of the target report in the **File Name** column.
3. Drag controls from the menu bar to the canvas. In this topic, the **Bar Chart** control on the **Tendency Chart** tab is dragged to the current report. You can drag a control from the menu bar to the canvas to use the control as a component in the report.



4. Click the **Bar Chart** component. Then, select a data store on the **Data Config** tab of the configuration section. If the required data store is added, click the name of the data store on the **Data Config** tab. If you need to add a data store, click **Add** on the **Data Config** tab. In the **Create Pivot Table** dialog box, set relevant parameters and click **OK**.

You can set the Choose Data parameter to Select a spreadsheet or Use External Data Store.

- o **Select a spreadsheet**: You can select an editable worksheet in a workbook of the current user as the data store.



Parameter	Description
Choose Data	The range of the data to be analyzed. Select Select a spreadsheet .
Spreadsheet	The workbook from which the data is analyzed. Select a workbook from the Spreadsheet drop-down list.

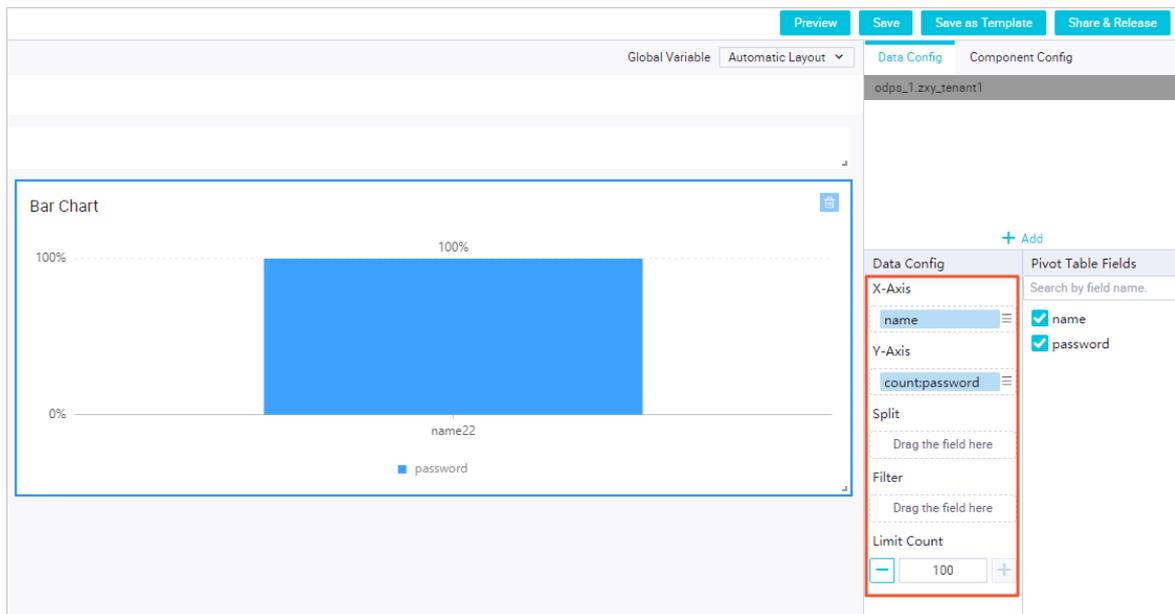
Parameter	Description
SHEET	The worksheet of which the data is analyzed. Select a worksheet from the SHEET drop-down list.

- o **Use External Data Store:** You can select a MySQL data store or the API of DataService Studio.

Parameter	Description
Choose Data	The range of the data to be analyzed. Select Use External Data Store .
Type	The type of the data source. Valid values: Mysql and Data Services .
Workspace	The workspace where the MySQL data store resides.
Data Store	The name of the connection to the data store. To create a connection to a data store, perform the following steps: Click the Workspace Manage icon in the upper-right corner. On the page that appears, click Data Source in the left-side navigation pane. On the Data Source page, create a connection to a data store. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note This parameter is available only when you set the Type parameter to Mysql.</p> </div>
Table	The table of which the data is analyzed. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note This parameter is available only when you set the Type parameter to Mysql.</p> </div>
API Group	The API group to which the API belongs. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note This parameter is available only when you set the Type parameter to Data Services.</p> </div>

Parameter	Description
API	<p>The API to be used as the data source.</p> <p>Note This parameter is available only when you set the Type parameter to Data Services.</p>

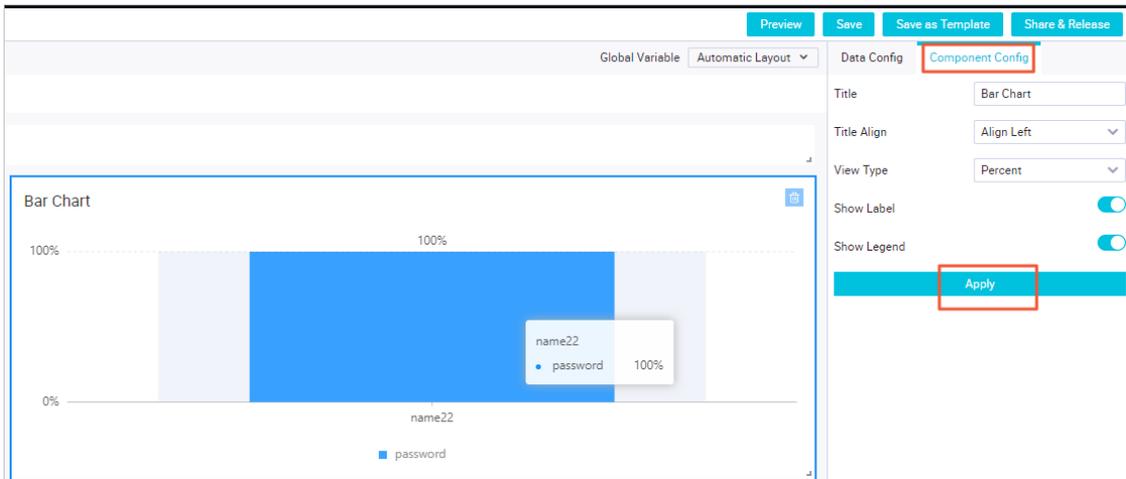
5. Select fields as statistical items. The fields that are required vary with the chart type. For example, you must specify the X-axis, Y-axis, split, and filter fields for a column chart by dragging fields from the **Pivot Table Fields** section to the **Data Config** section. You can also specify the number of vertical columns that can appear in the column chart.



Multiple charts can use the same data store. They can use the data store in different ways without affecting each other. A chart can use only one data store. When you click a chart and drag fields from the Pivot Table Fields section to the Data Config section, the chart is associated with the data store.

6. Configure the information about the column chart.
 - i. On the canvas, click the **Bar Chart** component.
 - ii. Click the **Component Config** tab in the right-side configuration section.

iii. On the **Component Config** tab, set relevant parameters.



Parameter	Description
Title	The title of the component.
Title Align	The alignment of the chart title. Valid values: Align Left , Align Center , and Align Right .
View Type	The display mode of vertical columns. Valid values: Stack , Parallel , and Percent .
Show Label	Specifies whether to display labels for the component.
Show Legend	Specifies whether to display legends for the component.

7. Click **Apply**.

8. Return to the **Report** page or preview, save, share, or release the report as needed.

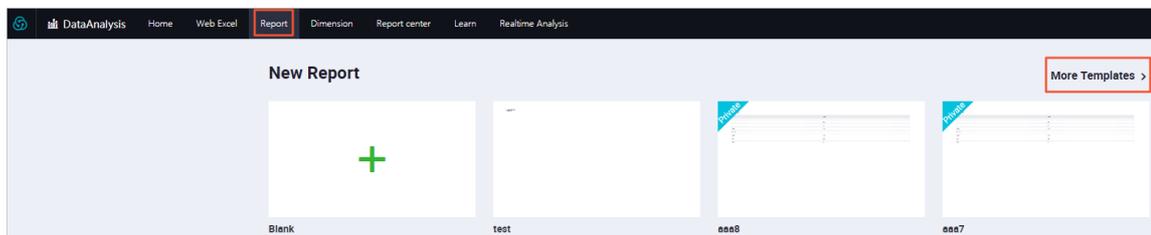
- **Return:** You can click **Return** in the upper-left corner to return to the **Report** page. On the Report page, you can view other reports and go to the report editing page of other reports.
- **Preview:** You can click **Preview** in the upper-right corner to preview the report.
- **Save:** You can click **Save** in the upper-right corner to save the report, so that you can open and edit the saved report next time.
- **Save as Template:** You can save a report as a template, so that you can create a report based on the template. Perform the following steps to save a report as a template:
 - a. On the report editing page, click **Save as Template** in the upper-right corner.
 - b. On the **Preview** page, click **Next Step (Template setup)**.

c. In the **Template settings** dialog box, set relevant parameters.

Parameter	Description
Type	Specifies whether to show or hide the template for other users. Valid values: Private and Open .
Name	The name of the template. The name can be up to 256 characters in length.
Description	The description of the template. The description can be up to 1,024 characters in length.

d. Click **OK**.

After you save the report as a template, you can click the template in the **New Report** section of the **Report** page to create a report.



- o **Share & Release**: You can click **Share & Release** in the upper-right corner to share and release the report. You can share the report with specified users or all users. If you need to share the report to specific users, click **Add** to specify the users.

2.7.5. Go to the report center

On the Data analysis report center page, you can view the visualized report solution that DataAnalysis provides.

Procedure

1. [Log on to the DataWorks console.](#)
2. On the **DataStudio** page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. In the top navigation bar of the **DataAnalysis** page, click **Report center** to go to the **Data analysis report center** page.

2.7.6. Go to the learning center

On the Data analytics Learning Center page, you can view relevant materials and learn about DataAnalysis.

Procedure

1. [Log on to the DataWorks console.](#)
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. In the top navigation bar of the DataAnalysis page, click **Learn** to go to the **Data analytics Learning Center** page.

2.8. Administration

2.8.1. Overview

Generally, developers need to test workflows and nodes on the Operation Center page.

As a key tool for routine O&M, Operation Center enables you to manage and maintain the workflows and nodes that you have committed. The Operation Center service consists of four modules: Dashboard, Nodes, Node Instances, and Monitor.

- **Dashboard:** enables you to view and manage all global nodes of DataWorks. It displays various information, including **Instances**, **Instances Run Today**, **Node Runtime**, **Instances Run in the Last Month**, **Nodes with Errors in the Last Month**, and **Node Types** of the current workspace.
- **Nodes:** provides **Recurring** and **Manually Triggered**.
- **Node Instances:** provides **Recurring**, **Manually Triggered**, **Smoke Test** and **Retroactive**. You can manage them in a list view or DAG.
 - The list view displays the running status of nodes in a list. You can add multiple alerts at a time, change owners, and add nodes to baselines.
 - In the DAG, you can maintain and manage the running status of nodes and their dependencies on ancestor and descendant nodes. You can also perform operations, such as retroactive data generation and rerun, for a single node.
- **Monitor:** provides **Baseline Instances**, **Baselines**, **Events**, **Alert Triggers**, and **Alerts**.

2.8.2. Dashboard

The Dashboard page provides information about the node running status, the trend of the number of nodes that were run, the node running time, and the nodes with errors.

Go to the Dashboard page

1. [Log on to the DataWorks console.](#)
2. Click  in the upper-left corner and choose **All Products > Operation Center**. By default, the **Dashboard** page appears.

The **Dashboard** page consists of the following sections: **Instances**, **Instances Run Today**, **Node Runtime**, **Nodes with Errors In the Last Month**, **Instances Run In the Last Month**, and **Node Types**.

Note

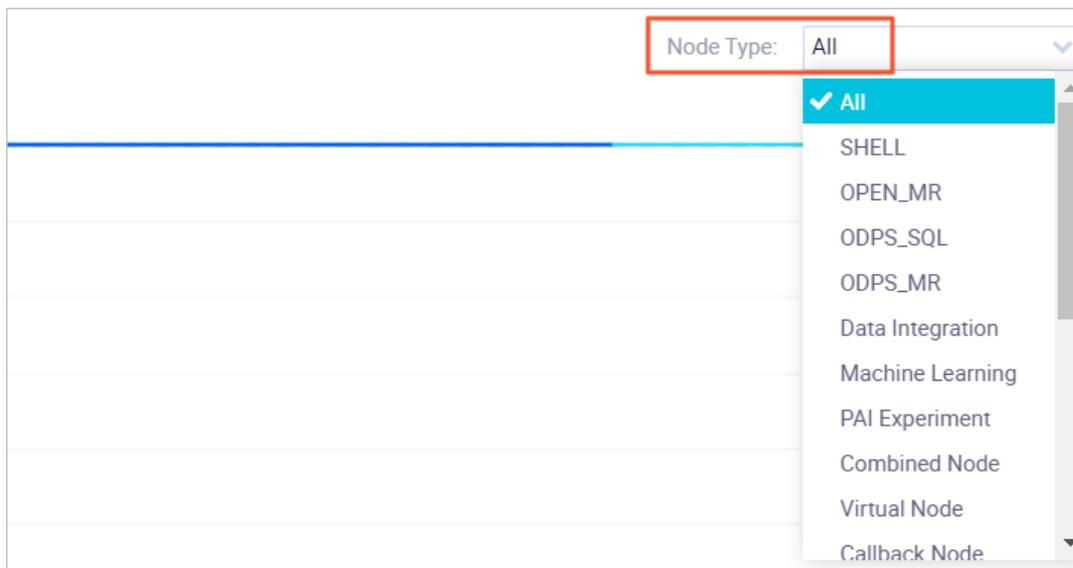
- Pending (Schedule): the instances whose scheduled time has not arrived. These instances will be automatically run when their scheduled time arrives.
- Pending (Resources): the instances whose scheduled time has arrived. These instances are waiting for scheduling resources.

View the summary of instance running

The **Instances** section displays the numbers of auto triggered node instances that were run today and yesterday, as well as the historical average. If the deviations among the three numbers are large, an exception occurred during a specific period of time. Further check and analysis are required.

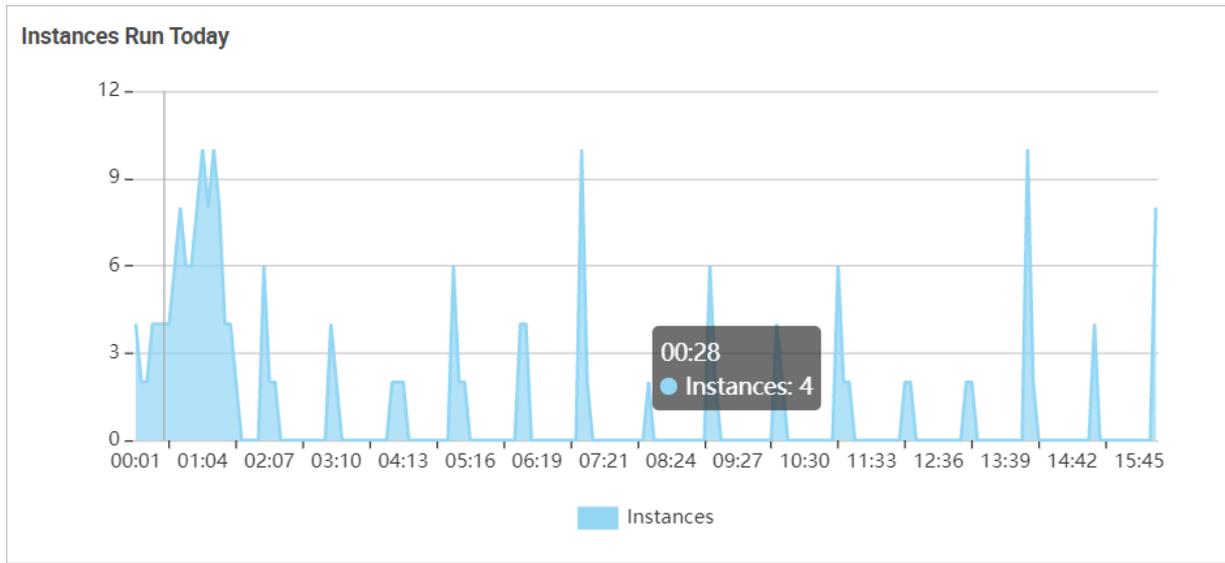
The statistical chart on the right of the Instances section shows three lines in different colors, representing the numbers of instances completed between 00:00 and 23:00 of today and yesterday, as well as the historical average. You can also view the number and proportion of instances in the pie chart on the left.

In the upper-right corner of the statistics chart, you can select a node type from the **Node Type** drop-down list to view statistics on the specified type of nodes.



View the statistics on instances run today

The **Instances Run Today** section displays the numbers of node instances that were running at different time points of the current day. In the chart, you can find the time point when peak concurrency occurred and the maximum number of concurrent nodes. Based on the information, you can determine whether to avoid node scheduling at the peak hours and adjust the scheduling time of nodes.



View the rankings of nodes based on their running time

The **Node Runtime** section displays nodes with a specified timestamp in the current workspace by their running time. By default, this section displays the top 10 nodes in descending order of their running time. You can view **Node ID**, **Node Name**, **Owner**, and **Runtime** of each node.

Node Runtime			
Node ID	Node Name	Owner	Runtime
[Node ID]	[Node Name]	[Owner]	1Hours19Min...
[Node ID]	[Node Name]	[Owner]	1Hours45Sec...

View the rankings of nodes with errors in the last month

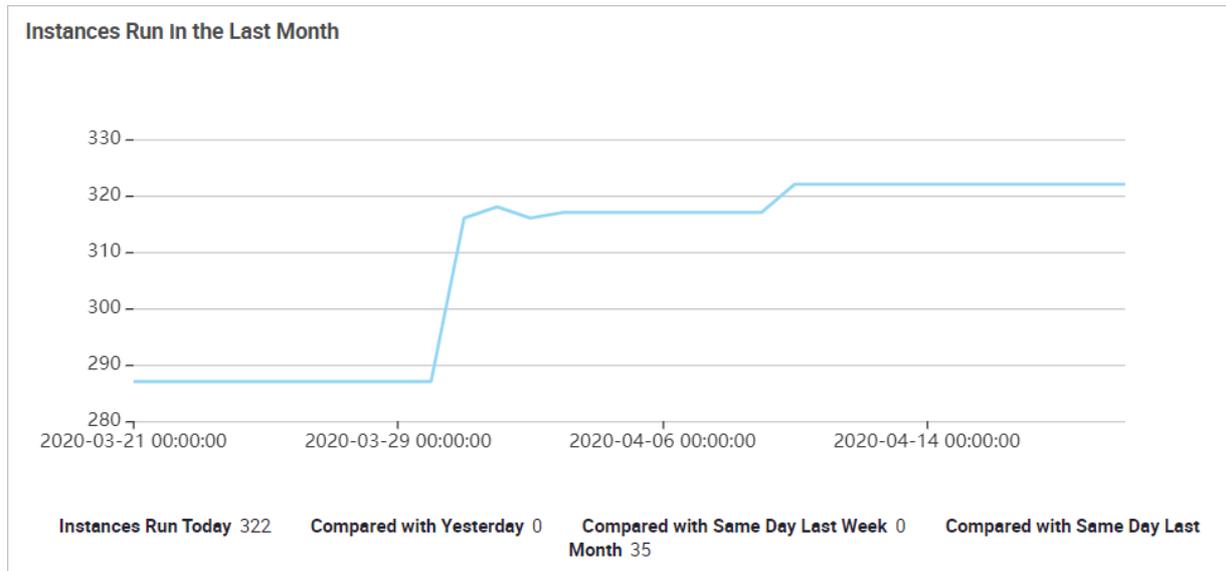
The **Nodes with Errors In the Last Month** section displays the top 10 nodes with the most errors in the last month. You can view **Node ID**, **Node Name**, **Owner**, and **Errors** of each node.

Nodes with Errors In the Last Month			
Node ID	Node Name	Owner	Errors
[Node ID]	[Node Name]	[Owner]	360
[Node ID]	[Node Name]	[Owner]	360
[Node ID]	[Node Name]	[Owner]	360
[Node ID]	[Node Name]	[Owner]	360

You can click a node ID to go to the node details page.

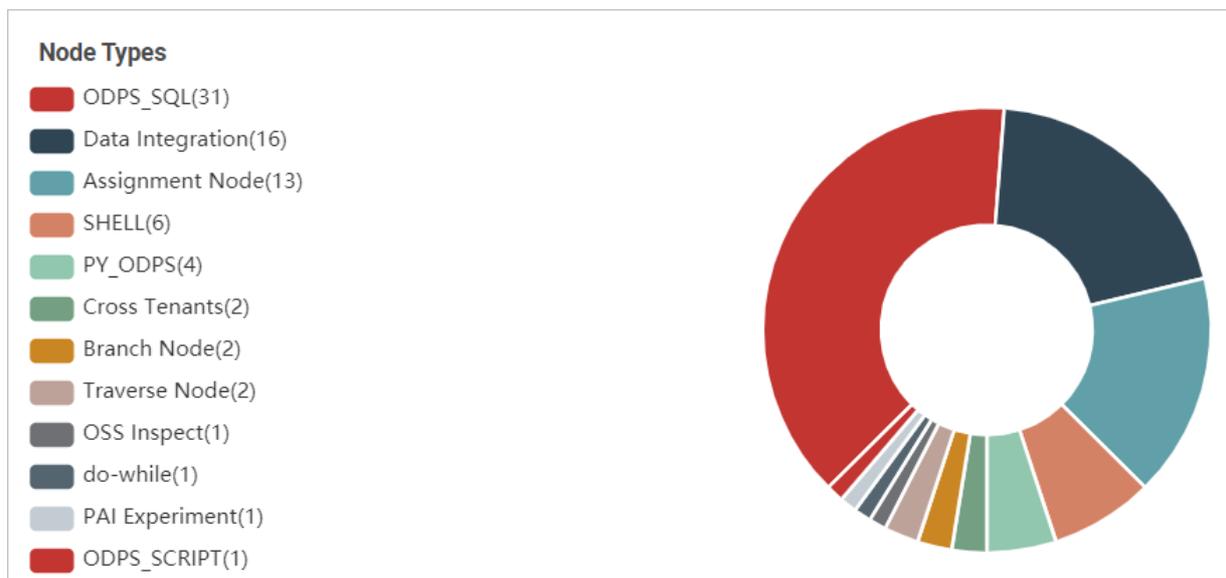
View the trend of the number of instances that were run

The **Instances Run In the Last Month** section displays the number of instances that were run today, comparison with yesterday, comparison with the same day last week, and comparison with the same day last month.



View the node distribution by node type

In the Node Types section, you can move the pointer over a sector of the pie chart to view the number and proportion of nodes of the specific type.



2.8.3. Manage real-time synchronization nodes

This topic describes how to manage real-time synchronization nodes on the Operation Center page.

Procedure

1. [Log on to the DataWorks console.](#)
2. On the **DataStudio** page, click the  icon in the upper-left corner and choose **All Products > Operation Center**.

3. In the left-side navigation pane of the Operation Center page, choose **RealTime Task Maintenance > Real Time DI**.
4. On the **Real Time DI** page, find your desired real-time synchronization node, click the node name, and then view the operations and maintenance (O&M) details about the node. On this page, you can **start**, **stop**, **undeploy**, or **configure alerting** for the real-time synchronization node.
 - o To start a node that is not running, perform the following steps:
 - a. Find the node and click **Start** in the Operation column.
 - b. In the **Start** dialog box, configure the parameters.

Parameter	Description
Whether to reset the site	Specifies whether to set the time point for the next startup. If you select Reset site , the Start time point and Time zone parameters are required.
Start time point	The date and time for starting the real-time synchronization node.
Time zone	The time zone where the source resides. Select a time zone from the Time zone drop-down list.
Failover	<ul style="list-style-type: none"> ■ The condition for automatically terminating the real-time synchronization node. You can specify the maximum number of dirty data records allowed. If you set the value to 0, no dirty data records are allowed. If the value is empty, the running of the node continues no matter whether dirty data records exist. ■ You can also specify the maximum number of failover times. If you do not specify Failover, the node is automatically terminated if the node fails 100 times within 5 minutes. This avoids resource occupation caused by frequent startup.

- c. Click **OK**.
- o To stop a running node, perform the following steps:
 - a. Find the node and click **Stop** in the Operation column.
 - b. In the message that appears, click **Stop**.
- o To undeploy a node that is not running, perform the following steps:
 - a. Find the node and click **Offline** in the Operation column.
 - b. In the message that appears, click **Offline**.
- o Find the node and click **Alarm settings** in the Operation column. On the page that appears, you can view alert event information and alert rules on the **Alert event** and **Alarm rules** tabs.
- o To configure alert settings for a node, perform the following steps:
 - a. Select your desired node and click **New Alarm** in the lower part of the page.

b. In the **New rule** dialog box, configure the parameters.

Parameter	Description
Name	Required. The name of the rule that you want to create.
Description	The description of the rule.
Indicators	The metrics in the rule that you want to create. Valid values: Task Status , Business latency , Failover , Dirty Data , and DDL error .
Threshold	The threshold for reporting an alert. The default value is 5 minutes for both WARNING and CRITICAL alerts.
Alarm interval	The interval at which an alert is reported. The default value is 5 minutes.
WARNING	The methods used to send alert notifications. Set the value to Mail .
CRITICAL	
Receiver	The alert recipient. Select a recipient from the Receiver drop-down list.

- c. Click **OK**.
- o To modify alert settings for a node, perform the following steps:
 - a. Select your desired node and click **Operation alarm** in the lower part of the page.
 - b. In the **Operation alarm** dialog box, specify the **Operation type** and **Alarm indicators** parameters.
DataWorks automatically modifies all the rules for the selected alert types at a time.
 - c. Click **OK**.

2.8.4. Auto triggered node O&M

2.8.4.1. Manage auto triggered nodes

Auto triggered nodes are automatically run as scheduled after they are committed to the scheduling system.

Note

- By default, the auto triggered node list displays nodes in all the workflows created by the current account.
- After you commit a node, the scheduling system automatically generates and runs an instance of the node at 23:30 the next day. If you commit a node after 23:30, the scheduling system generates and runs an instance of the node on the third day.
- Do not perform any operations on the **Workspace name_root** node, which is the root node of the workspace. All instances of auto triggered nodes depend on this node. If this node is frozen, instances of auto triggered nodes cannot be run.

Manage auto triggered nodes in the node list

The **Cycle Task** page displays auto triggered nodes that are committed to the scheduling system in a list.

Operation	Description
Filter	<p>Find required nodes by setting parameters in the red box marked with 1 in the preceding figure. You can search for nodes by node name or node ID and set the Node Type, Owner, My Nodes, Modified Today, and Frozen parameters to filter nodes.</p> <p> Note When you search for nodes by node name, the search result is affected by other filter conditions you specified. Only the nodes that meet both the specified search condition and other filter conditions are returned in the search result.</p>
DAG	Click DAG in the Actions column of a node to view the directed acyclic graph (DAG) of the node. You can view the node information, such as properties, operations logs, and code, in the DAG.
Test	Click Test in the Actions column of a node to test the node. For more information, see Manage test instances .
Patch Data	Click Patch Data in the Actions column of a node and select an item from the drop-down list to generate retroactive data for the node. For more information, see Manage retroactive instances .
More operations	<p>Click More in the Actions column of a node to perform more operations on the node. You can perform the following operations on the node:</p> <ul style="list-style-type: none"> • Select Freeze to freeze the node. After the node is frozen, DataWorks can still generate instances of the node, but does not run the instances of the node and its descendant instances. • Select Unfreeze to unfreeze the node. After the node is unfrozen, DataWorks can normally run the instances of the node and its descendant instances. • Select View Instances to view the instances of the node. • Select Configure Alert Trigger to configure alert triggers for the node. • Select Change Owner to change the owner of the node. • Select Add to Baseline to add the node to a baseline. • Select Change Resource Group to change the resource group used to run the node if multiple resource groups exist in the workspace. • Select Configure Data Quality Rules to configure rules for monitoring the data quality of the node. • Select View Lineage to view the lineage of the node. • Select View Ancestor and Descendant Nodes to go to the Node Information page, where you can view node information on the Ancestor Nodes and Descendant Nodes tabs.
Batch operations	Select multiple nodes and click a button in the red box marked with 3 in the preceding figure to perform batch operations. The buttons include Configure Alert Trigger , Change Owner , Change Resource Group , Add to Baseline , Freeze , Unfreeze , and Delete .

Manage auto triggered nodes in a DAG

Click the name of a node or **DAG** in the Actions column to view the DAG of the node. In the DAG, you can right-click the node to perform related operations.

Operation	Description
Show Ancestor Nodes or Show Descendant Nodes	View ancestor or descendant nodes at one or more levels. If the workflow contains three or more nodes, the DAG displays only the current node and hides its ancestor and descendant nodes.

Operation	Description
View Node Details	Go to the Node Information page to view the node information, including the input table, output table, ancestor nodes, and descendant nodes.
View Code	View the code of the node.
Edit Node	Go to the DataStudio page to modify the node.
View Instances	View the instances of the node.
View Lineage	View the lineage of the node.
Test	Go to the Smoke Test dialog box. Set the Smoke Test Instance Name and Data Timestamp parameters and click OK . Then, the Test Instance page appears.
Run	Select Current Node Retroactively , Current and Descendant Nodes Retroactively , or Mass Nodes Retroactively .
Freeze	Pause the scheduling of the node.
Unfreeze	Resume the scheduling of the node after it is frozen.
Configure Data Quality Rules	Configure rules for monitoring the data quality of the node.

2.8.4.2. Manage auto triggered node instances

Auto triggered node instances are snapshots taken for auto triggered nodes that are run.

An instance is generated each time when an auto triggered node is run as scheduled. You can manage auto triggered node instances. For example, you can view the running status of instances, and stop, rerun, and unfreeze instances.

 **Note** DataWorks regularly generates instances for auto triggered nodes. Each generated instance runs the latest code. If you modify and recommit the node code after instances are generated, the instances that have not been run will run the latest code.

Manage auto triggered node instances in the instance list

You can manage auto triggered node instances in the instance list. For example, you can check operational logs, rerun instances, and stop running instances.

Operation	Description
Filter	<p>Find required instances by setting parameters in the red box marked with 1 in the preceding figure.</p> <p>You can search for instances by node name or node ID and set the following parameters to filter instances: Data Timestamp, Node Type, Run At, Solution, Workflow, Region, Engine type, Engine instance, Baseline, Owner, Recurrence, Status, My Nodes, My Nodes with Errors, My Incomplete Nodes, and Re-run node.</p> <p> Note By default, the data timestamp is set to the previous day of the current day.</p>

Operation	Description
Stop	Stop the instance. You can stop instances only when they are in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.
Rerun	Rerun the instance. After the instance is rerun, its descendant instances will be run as scheduled. Perform this operation if an instance fails or it is not run as scheduled. <div style="background-color: #e6f2ff; padding: 5px;"> <p> Note This operation applies only to instances in the Pending (Ancestor), Successful, or Failed state.</p> </div>
Rerun Descendant Nodes	Rerun the instance and its descendant instances. You must specify the instances to rerun. After they are run, their descendant instances will be run as scheduled. This operation applies to data recovery. <div style="background-color: #e6f2ff; padding: 5px;"> <p> Note You can select instances only in the Pending (Ancestor), Successful, or Failed state. The value No appears in the Meet Rerun Condition column of instances in other states, and you cannot select the instances.</p> </div>
Set Status to Successful	Set the status of the instance to Successful and run its descendant instances as scheduled. You can perform this operation if an instance fails. <div style="background-color: #e6f2ff; padding: 5px;"> <p> Note This operation applies only to failed instances but not workflows.</p> </div>
Freeze	Freeze the instance if it is in the Running state.
Unfreeze	Unfreeze the instance after it is frozen. <ul style="list-style-type: none"> • If the instance is not run, DataWorks automatically runs this instance after its ancestor instances are run. • If all the ancestor instances have been run, the instance is directly set to Failed. You must manually rerun the instance.
Batch operations	Select multiple instances and click a button in the red box marked with 3 in the preceding figure to perform batch operations. The buttons include Stop , Rerun , Set Status to Successful , Freeze , and Unfreeze .

Manage auto triggered node instances in a DAG

Click the name of an instance or **DAG** in the **Actions** column to view the directed acyclic graph (DAG) of the instance. In the DAG, you can right-click the instance to perform related operations.

Operation	Description
Show Ancestor Nodes or Show Descendant Nodes	View ancestor or descendant instances at one or more levels. If a workflow contains three or more instances, the DAG displays only the current instance and hides its ancestor and descendant instances.
View Runtime Log	View the operational logs of the instance if it is in the Running , Successful , or Failed state.
View Code	View the code of the instance.

Operation	Description
Edit Node	Go to the DataStudio page to modify the node to which the instance belongs.
View Lineage	View the lineage of the instance.
More operations	View more instance information on the General, Context, Runtime Log, Operation Log, and Code tabs.
Stop	Stop the instance if it is in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.
Rerun	<p>Rerun the instance. After the instance is rerun, its descendant instances will be run as scheduled. Perform this operation if an instance fails or it is not run as scheduled.</p> <p> Note This operation applies only to instances in the Pending (Ancestor), Successful, or Failed state.</p>
Rerun Descendant Nodes	<p>Rerun the instance and its descendant instances. You must specify the instances to rerun. After they are run, their descendant instances will be run as scheduled. This operation applies to data recovery.</p> <p> Note You can select instances only in the Pending (Ancestor), Successful, or Failed state. The value No appears in the Meet Rerun Condition column of instances in other states, and you cannot select the instances.</p>
Set Status to Successful	<p>Set the status of the instance to Successful and run its descendant instances as scheduled. You can perform this operation if an instance fails.</p> <p> Note This operation applies only to failed instances but not workflows.</p>
Resume	Continue to run the instance after it fails.
Emergency Operations	<p>Perform emergency operations, including Delete Dependencies, Change Priority, and Force Rerun. Perform these operations in emergency only. These operations take effect only on the current instance for one time.</p> <p>Select Delete Dependencies to delete dependencies of the current instance. You can perform this operation so that you can start the current instance when the ancestor instances fail and the current instance does not depend on the data of the ancestor instances.</p>
Freeze	Freeze the instance if it is in the Running state.
Unfreeze	<p>Unfreeze the instance after it is frozen.</p> <ul style="list-style-type: none"> • If the instance is not run, DataWorks automatically runs this instance after its ancestor instances are run. • If all the ancestor instances have been run, the instance is directly set to Failed. You must manually rerun the instance.

2.8.4.3. Manage retroactive instances

DataWorks runs retroactive instances to generate retroactive data for auto triggered nodes. You can manage retroactive instances. For example, you can view the running status of instances, and stop, rerun, or unfreeze instances.

Limits

- When DataWorks generates retroactive data in a period of time for a node, if one instance of the node fails on a day, the retroactive instance for that day is also set to Failed. DataWorks will not run instances of this node for the next day. To sum up, DataWorks runs instances of a node on a day only when all its instances of the previous day are successful.
- For a self-dependent auto triggered node, if the first instance for which retroactive data needs to be generated has a last-cycle instance on the previous day but the last-cycle instance is not run, the retroactive instance cannot be triggered. If the first instance for which retroactive data needs to be generated does not have a last-cycle instance on the previous day, the retroactive instance is directly triggered.
- DataWorks generates alerts only for auto triggered node instances.
- If an auto triggered node has an instance in the Running state, its retroactive or test instance can start to run only after this auto triggered node instance is run.
- If both an auto triggered node instance and a retroactive instance are running for a node at the same time, you must stop the retroactive instance to ensure proper running of the auto triggered node instance.

Go to the page for configuring retroactive instances

1. [Log on to the DataWorks console.](#)
2. Click  in the upper-left corner and choose **All Products > Operation Center**.
3. In the left-side navigation pane, choose **Cycle Task Maintenance > Cycle Task**.
4. On the page that appears, click the rightward arrow in the middle to show the Actions column in the node list. Find the target node, click **Patch Data** in the Actions column, and then select a mode for generating retroactive data.

You can also right-click the target node in the directed acyclic graph (DAG), move the pointer over **Run**, and then select a mode for generating retroactive data.

Generate retroactive data for the current node

1. Find the target node, click **Patch Data** in the Actions column, and then select **Current Node Retroactively**.
2. In the **Patch Data** dialog box, set the parameters.

Parameter	Description
Retroactive Instance Name	The name of the retroactive instance. DataWorks automatically generates a name, which can be changed.
Data Timestamp	The data timestamp of the retroactive instance.
Node	The name of the node for which you want to generate retroactive data. You cannot change the name.

Parameter	Description
Parallelism	<p>Specifies whether to generate multiple retroactive instances at a time.</p> <ul style="list-style-type: none"> ◦ If you select Disable, only one retroactive instance is generated. The retroactive instance is run multiple times based on the data timestamp in sequence. ◦ If you select 2 Parallel Groups, 3 Parallel Groups, 4 Parallel Groups, or 5 Parallel Groups, multiple retroactive instances are generated. The retroactive instances are run based on the data timestamp in parallel or in sequence. <ul style="list-style-type: none"> ▪ If the number of days in the data timestamp is smaller than the number of parallel groups, the retroactive instances are run in parallel. For example, the data timestamp is from January 11 to January 13, and you select 4 Parallel Groups. In this case, three retroactive instances are generated for each day in the data timestamp, and the three retroactive instances are run in parallel. ▪ If the number of days in the data timestamp is greater than the number of parallel groups, some instances must be run multiple times in sequence while others are run in parallel. For example, the data timestamp is from January 11 to January 13, and you select 2 Parallel Groups. In this case, two retroactive instances are generated. They are run in parallel for once, and one of them must be run for a second time.

3. Click **OK**.

Generate retroactive data for the current and descendant nodes

1. Find the target node, click **Patch Data** in the Actions column, and then select **Current and Descendant Nodes Retroactively**.
2. In the **Patch Data** dialog box, set the parameters, including **Nodes**.

Parameter	Description
Retroactive Instance Name	The name of the retroactive instance. DataWorks automatically generates a name, which can be changed.
Data Timestamp	The data timestamp of the retroactive instance.
Parallelism	<p>Specifies whether to generate multiple retroactive instances at a time.</p> <ul style="list-style-type: none"> ◦ If you select Disable, only one retroactive instance is generated. ◦ If you select 2 Parallel Groups, 3 Parallel Groups, 4 Parallel Groups, or 5 Parallel Groups, multiple retroactive instances are generated.
Nodes	The nodes for which you want to generate retroactive data. You can set Node Name and Node Type to filter nodes.

3. Click **OK**.

Generate retroactive data for a large number of nodes

1. Find the target node, click **Patch Data** in the Actions column, and then select **Mass Nodes Retroactively**.

2. In the **Patch Data** dialog box, set the parameters. The following table describes the parameters for generating retroactive data for a large number of nodes.

Parameter	Description
Retroactive Instance Name	The name of the retroactive instance. DataWorks automatically generates a name, which can be changed.
Data Timestamp	<p>The data timestamp of the retroactive instance.</p> <p> Note We recommend that you do not set this parameter to a long range. Otherwise, the retroactive instance may be delayed due to insufficient resources.</p>
Parallelism	<p>Specifies whether to generate multiple retroactive instances at a time.</p> <ul style="list-style-type: none"> ◦ If you select Disable, only one retroactive instance is generated. ◦ If you select 2 Parallel Groups, 3 Parallel Groups, 4 Parallel Groups, or 5 Parallel Groups, multiple retroactive instances are generated.
Nodes	<ul style="list-style-type: none"> ◦ If you select the Current Node check box, retroactive instances are generated for the current node and its descendant nodes. ◦ If you clear the Current Node check box, the current node is dry-run and retroactive instances are generated for its descendant nodes.
Workspaces	The workspaces of the nodes for which DataWorks needs to generate retroactive data. Select workspaces under Available Workspaces and add them to Selected Workspaces . Fuzzy match is supported when you search for workspaces under Available Workspaces.
Node Whitelist	<p>The nodes outside the selected workspaces, for which DataWorks needs to generate retroactive data.</p> <p> Note You can search for nodes by node ID.</p>
Node Blacklist	<p>The nodes inside the selected workspaces, for which DataWorks does not need to generate retroactive data.</p> <p> Note You can search for nodes by node ID.</p>

3. Click **OK**.

Manage retroactive instances in the instance list

Operation	Description
-----------	-------------

Operation	Description
Filter	<p>Find required instances by specifying filter conditions.</p> <p>You can search for instances by node name or node ID and set the following parameters to filter instances: Retroactive Instance Name, Node Type, Owner, Run At, Data Timestamp, Engine type, Engine instance, Baseline, and My Nodes.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note By default, the data timestamp is set to the previous day of the current day.</p> </div>
DAG	View the DAG of the instance. You can view the running result of the instance in the DAG.
Stop	Stop the instance. You can stop instances only when they are in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.
Rerun	Rerun the instance.
Rerun Descendant Nodes	Rerun the descendant instances of the instance.
Freeze	Pause the scheduling of the instance.
Recover	Resume the scheduling of the instance after it is paused.
View Lineage	View the lineage of the instance.

Manage retroactive instances in a DAG

Click the name of an instance or **DAG** in the Actions column to view the DAG of the instance. In the DAG, you can right-click the instance to perform related operations.

 **Note** After you click **Refresh** in the upper-right corner, the DAG of the instance is refreshed, but the operational logs are not.

Operation	Description
Show Ancestor Nodes or Show Descendant Nodes	Show ancestor or descendant instances at one or more levels. If a workflow contains three or more instances, the DAG displays only the current instance and hides its ancestor and descendant instances.
View Runtime Log	View the operational logs of the instance if it is in the Running, Successful, or Failed state.
View Code	View the code of the instance.
Edit Node	Go to the DataStudio page to modify the node to which the instance belongs.
View Lineage	View the lineage of the instance.

Operation	Description
Stop	Stop the instance. You can stop instances only when they are in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.
Rerun	Rerun the instance if it is in the Failed or an abnormal state.
Rerun Descendant Nodes	Rerun all the descendant instances of the instance.
Set Status to Successful	Set the status of the instance to Successful and run its descendant instances as scheduled. You can perform this operation if an instance fails. <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> ? Note This operation applies only to failed instances but not workflows. </div>
Emergency Operations	Perform emergency operations in emergency only. Emergency operations take effect only on the current instance for one time. Select Delete Dependencies to delete dependencies of the current instance. You can perform this operation so that you can start the current instance when the ancestor instances fail and the current instance does not depend on the data of the ancestor instances.
Freeze	Pause the scheduling of the instance.
Unfreeze	Resume the scheduling of the instance after it is paused.

2.8.4.4. Manage test instances

Test instances are generated when you test auto triggered nodes. You can manage test instances.

Go to the Test Instance page

1. [Log on to the DataWorks console.](#)
2. Click  in the upper-left corner and choose **All Products > Operation Center**.
3. In the left-side navigation pane, choose **Cycle Task > Test Instance**. The Test Instance page appears, where you can view the list and directed acyclic graphs (DAGs) of test instances.

Manage test instances in the instance list

You can manage test instances in the instance list. For example, you can rerun, freeze, or unfreeze instances, set the status of instances to Successful, view the lineage of instances, and check operational logs.

Operation	Description
Filter	Find required instances by specifying filter conditions. You can search for instances by node name or node ID and set the following parameters to filter instances: Node Type , Owner , Run At , Data Timestamp , Status , Region , Engine type , Engine instance , Baseline , My Nodes , Tested by Me Today , and Frozen .
Stop	Stop the instance. You can stop instances only when they are in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.

Operation	Description
Rerun	<p>Rerun the instance. After the instance is rerun, its descendant instances will be run as scheduled. Perform this operation if an instance fails or it is not run as scheduled.</p> <p> Note This operation applies only to instances in the Pending (Ancestor), Successful, or Failed state.</p>
More operations	Click More in the Actions column and then select an operation. The operations include Set Status to Successful , Freeze , Unfreeze , View Lineage , and View Runtime Log .
Batch operations	Select multiple instances and click a button in the lower part of the instance list to perform batch operations. The buttons include Stop , Rerun , Set Status to Successful , Freeze , and Unfreeze .

Manage test instances in a DAG

Click the name of an instance or **DAG** in the Actions column to view the DAG of the instance. In the DAG, you can right-click the instance to perform related operations.

Operation	Description
View Runtime Log	View the operational logs of the instance if it is in the Running , Successful , or Failed state.
View Code	View the code of the instance.
Edit Node	Go to the DataStudio page to modify the node to which the instance belongs.
View Lineage	View the lineage of the instance.
Stop	Stop the instance. You can stop instances only when they are in the Pending (Resource) or Running state. After you perform this operation, the instance enters the Failed state.
Rerun	<p>Rerun the instance. After the instance is rerun, its descendant instances will be run as scheduled. Perform this operation if an instance fails or it is not run as scheduled.</p> <p> Note This operation applies only to instances in the Pending (Ancestor), Successful, or Failed state.</p>
Set Status to Successful	<p>Set the status of the instance to Successful and run its descendant instances as scheduled. You can perform this operation if an instance fails.</p> <p> Note This operation applies only to failed instances but not workflows.</p>
Freeze	Pause the scheduling of the instance.

Operation	Description
Unfreeze	<p>Resume the scheduling of the instance after it is frozen.</p> <ul style="list-style-type: none"> • If the instance is not run, DataWorks automatically runs this instance after its ancestor instances are run. • If all the ancestor instances have been run, the instance is directly set to Failed. You must manually rerun the instance.

2.8.5. Manually triggered node O&M

2.8.5.1. Manage manually triggered nodes

Manually triggered nodes are nodes whose scheduling type is set to manual before they are committed to the scheduling system.

 **Note** After a manually triggered node is committed to the scheduling system, it will run only after it is manually triggered.

Manage manually triggered nodes in the node list

The manually triggered node list displays manually triggered nodes that are committed.

Operation	Description
Filter	<p>Find required nodes by specifying filter conditions.</p> <p>You can search for nodes by node name and set the Owner, My Nodes, and Modified Today parameters to filter nodes.</p> <p> Note When you search for nodes by node name, the search result is affected by other filter conditions you specified. Only nodes that meet both the specified search condition and other filter conditions are returned in the search result.</p>
DAG	Click DAG in the Actions column of a node to view the directed acyclic graph (DAG) of the node. You can view the node information, such as the code and lineage, in the DAG.
Run	Click Run in the Actions column of a node to run the node. Manually triggered node instances are generated.
View Instances	Click View Instances in the Actions column of a node to go to the Manual Instance page, where you can view the instances of the node.
More operations	Choose More > Change Owner in the Actions column of a node to change the owner of the node.
Batch operations	Select multiple nodes and click Change Owner in the lower part of the page to change the owner of the nodes.

Manage manually triggered nodes in a DAG

Click the name of a node or **DAG** in the Actions column to view the DAG of the node. In the DAG, you can right-click the node to perform related operations.

Operation	Description
View Code	View the code of the node.
Edit Node	Go to the DataStudio page to modify the node.
View Instances	View the instances of the node.
View Lineage	View the lineage of the node.
Run	Run the node. After you click Run, manually triggered node instances are generated.
Change Resource Group	Change the resource group where the node is run.

2.8.5.2. Manage manually triggered node instances

DataWorks generates manually triggered node instances from manually triggered nodes. Manually triggered nodes do not have node dependencies. They must be run manually.

 **Notice** DataWorks generates alerts only for auto triggered node instances when they fail to be run.

Go to the page for managing manually triggered node instances

1. [Log on to the DataWorks console.](#)
2. Click  in the upper-left corner and choose **All Products > Operation Center**.
3. In the left-side navigation pane, choose **Trigger Task Maintenance > Trigger Instance**. The Trigger Instance page appears, where you can view the list and directed acyclic graphs (DAGs) of manually triggered instances.

Manage manually triggered node instances in the instance list

Operation	Description
Filter	Find required instances by specifying filter conditions. You can search for instances by instance name and set the Owner , Data Timestamp , and Run At parameters to filter instances.
DAG	Click DAG in the Actions column to view the DAG of the instance. You can view the running result of the instance in the DAG.
Stop	Click Stop to stop the instance if it is in the Running state.
Rerun	Rerun the instance.
Batch stop	Select multiple instances and click Stop to stop the selected instances at a time.

Manage manually triggered node instances in a DAG

Click the name of an instance or DAG in the Actions column to view the DAG of the instance. In the DAG, you can right-click the instance to perform related operations.

 **Note** A manually triggered node instance does not have dependencies, so only the current instance appears in the DAG.

Operation	Description
View Runtime Log	View the operational logs of the instance if it is in the Running, Successful, or Failed state.
View Code	View the code of the instance.
Edit Node	Go to the DataStudio page to modify the node to which the instance belongs.
View Lineage	View the lineage of the instance.
Stop	Stop the instance.
Rerun	Rerun the instance if it is in the Failed or an abnormal state.

2.8.6. MaxCompute engine O&M

DataWorks Operation Center allows you to view the jobs, quota groups, and projects of MaxCompute.

Prerequisites

A MaxCompute engine is added on the **Project Management** page.

Go to the MaxCompute page

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Operation Center**.
3. In the left-side navigation pane, choose **Engine Maintenance > MaxCompute**. The **Job Queue** tab appears by default. You can view the jobs, quota groups, and projects of MaxCompute.

View jobs

On the **Job Queue** tab, you can view the total number of jobs and numbers of jobs in the **Running**, **Waiting for Resources**, and **Initializing** states.

You can set the **Quota Group** and **Project** parameters to filter jobs. You can view the following parameters of each job: **Instance**, **Execution Time**, **CPU Usage (Minimum/Maximum)**, **Memory Usage (Minimum/Maximum)**, **Priority**, **Node**, **Submitted By**, **Project**, **Type**, **Quota Group**, **Cluster**, **Status**, and **Start Time**.

View quota groups

On the **MaxCompute** page, click the **Quotas** tab.

On the **Quotas** tab, you can view the following parameters of each quota group: **Project**, **Quota Group**, **Default**, **Cluster**, **Minimum CPU (cores)**, **Maximum CPU (cores)**, **Minimum Memory (bytes)**, **Maximum Memory (bytes)**, and **Projects**.

You can click the name of a quota group to view the resource usage information about the quota group.

View projects

On the **MaxCompute** page, click the **Projects** tab.

On the **Projects** tab, you can view the following parameters of each project: **Project**, **Owner**, **Cluster**, **Quota Group**, **Storage Used**, **Storage Quota**, **Storage Usage**, and **Files**.

2.8.7. Monitor

2.8.7.1. Overview

The Monitor module is a node monitoring and analysis system of DataWorks. Based on monitoring rules and node running status, the Monitor module determines whether, when, and how to trigger an alert, and whom an alert is sent to. It automatically selects the most appropriate alerting time, notification methods, and recipients.

The Monitor module provides you with the following benefits:

- Improves your efficiency on configuring monitoring rules.
- Prevents invalid alerts from bothering you.
- Automatically covers all important nodes for you.

General monitoring systems cannot meet the requirement of DataWorks. The reasons are as follows:

- DataWorks has numerous nodes, so it is difficult for you to find out the nodes to be monitored. Some DataWorks businesses have a large number of nodes, and dependencies between the nodes are complex. Even if you know the most important node, it is difficult to find all ancestor nodes of the node and monitor them all. In this case, if you simply monitor all nodes, a large number of invalid alerts may be generated. In consequence, you may miss those useful alerts.
- The alerting method varies with nodes. For example, some monitoring tasks require the relevant nodes to run for more than one hour before triggering alerts, while other monitoring tasks require the relevant nodes to run for more than two hours. It is extremely complex to set a monitoring node for each node, and it is difficult to predict the alert threshold value for each node.
- The alerting time varies with nodes. For example, an alert for an unimportant node can be sent after you start working in the morning. An alert for an important node needs to be sent immediately when an error occurs. General monitoring systems cannot tell the importance of each node.
- Different alerts require different operations to turn off.

The Monitor module provides comprehensive monitoring and alerting logic. You only need to provide the node name of your business. Then, the Monitor module automatically monitors the entire process of your node and creates standard alert triggers for the node. In addition, you can customize alerting triggers by completing basic settings.

Currently, the Monitor module has been used for monitoring all important businesses of Alibaba Group. Its full-path monitoring function guarantees the overall data output of all important businesses of Alibaba Group. In addition, it supports analyzing ancestor and descendant node paths to promptly detect risks and provide O&M advice for business departments. These functions of the Monitor module have guaranteed the long-term high stability of businesses in Alibaba Group.

2.8.7.2. Feature description

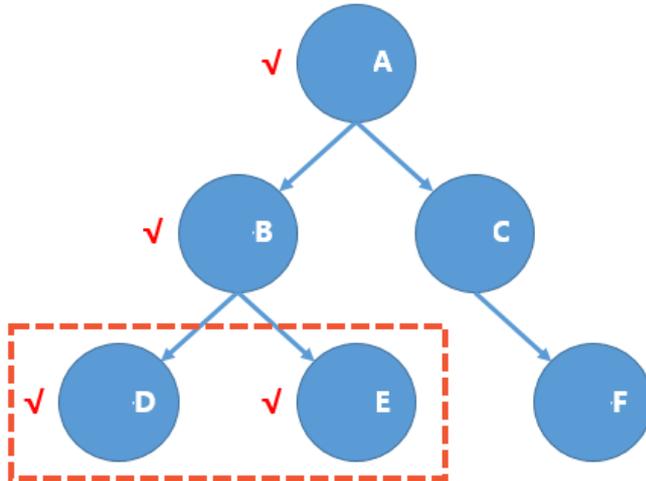
2.8.7.2.1. Baseline alert and event alert

This topic describes the functional logic of baseline alerts and event alerts from the aspects of monitoring scope, node capturing, alert object judgment, alerting time judgment, notification methods, and alert escalation.

Monitoring scope

A baseline is a management unit of a group of nodes, that is, a node group. You can specify nodes to monitor in a baseline.

After a baseline is monitored, all nodes of the baseline and its ancestor nodes are monitored. The Monitor module does not monitor all nodes by default. A node is monitored only when it has descendant nodes that are added to a monitoring baseline. If no descendant nodes are added to a monitoring baseline, the Monitor module does not report any alert even if the node fails.



As shown in the preceding figure, assume that DataWorks has only six nodes, and nodes D and E belong to a monitoring baseline. Nodes D and E and all their ancestor nodes are monitored by the Monitor module. That is, any error or slowdown on node A, B, D, or E will be detected by the Monitor module. However, nodes C and F are not monitored by the Monitor module.

Node capturing

After the nodes to be monitored are specified, if a monitored node incurs an exception, the Monitor module generates an event. All alert decisions are based on the analysis of this event. Two types of node exceptions are available. You can choose **Events > Event Type** to view them.

- **Error:** indicates that a node fails to run.
- **Slow:** indicates that the running time of a node is significantly longer than the average running time of the node in the past periods.

Note If a node times out and then encounters an error, two events are generated.

Alerting time judgment

Buffer, an important concept in the Monitor module, refers to the maximum time period that a node can be delayed. The latest start time of a node is obtained by subtracting the average uptime from the baseline time.

The baseline time of baseline A is 05:00, you must set the latest start time of node E to 04:10. This time is calculated by subtracting the average uptime of node F (20 minutes) and node E (30 minutes) from the baseline time 05:00. This time is also the latest completion time of node B in baseline A.

To ensure that the baseline time of baseline B is 06:00, you must set the latest completion time of node B to 04:00. This time, which is earlier than 04:10, is calculated by subtracting the average uptime of node D (2 hours) from the baseline time 06:00. To meet the baseline time of both baseline A and baseline B, you must set the latest completion time of node B to 04:00.

The latest completion time of node A is 02:00, which is calculated by subtracting the average uptime of node B (2 hours) from 04:00. The latest start time of node A is 01:50, which is calculated by subtracting the average uptime of node A (10 minutes) from 02:00. If node A fails to run before 01:50, it is probable that baseline A is broken.

If node A fails to run at 01:00, its buffer is 50 minutes, which is the difference between 01:00 and 01:50. As demonstrated in this example, buffer reflects the degree of caution for a node exception.

Baseline alert

Baseline alerting is an additional feature developed for baselines that are enabled. Each baseline must provide an alert buffer and committed time. Baseline alerting is the action of notifying the preset alert recipient three times at the interval of 30 minutes when the baseline completion time estimated by the Monitor module exceeds the alert buffer.

Notification method

Currently, baseline alerts are sent to the baseline owner by default. On the **Alert Triggers** page, you can find **Global Baseline Alert Trigger**, click **View Details**, and change the alert trigger method and the alerting action.

Gantt chart function

The Gantt chart function reflects the key path of a node. The function is provided by the **Baseline Instances** module of Monitor.

 **Note** The key path is the slowest upstream link that causes the node to be completed at this time point.

2.8.7.2.2. Custom alert trigger

Alert trigger customization is a lightweight monitoring function of the Monitor module.

You can customize all monitoring alert triggers by setting the following parameters:

- **Objects:** You can specify nodes, baselines, and workspaces as objects.
- **Trigger Condition:** Valid values include Completed, Uncompleted, Error, Uncompleted Cycle, and Overtime.
- **Notification Method:** Valid values include SMS and Email.
- **Maximum Alerts:** This parameter indicates the maximum number of alert reporting times. If the number of alerting times exceeds the preset threshold, no alerts are generated.
- **Minimum Alert Interval:** This parameter indicates the minimum time interval at which DataWorks reports alerts.
- **Quiet Hours:** This parameter indicates the specified period during which no alerts are reported.
- **Recipient:** This parameter indicates the person who receives alerts. You can set this parameter to the node owner or another recipient.

A monitoring rule uses the following five alert trigger conditions: Completed, Uncompleted, Error, Uncompleted Cycle, and Overtime.

- **Completed**

A completion alert can be set for nodes, baselines, and workspaces. Once all nodes of the preset objects are completed, the completion alert is reported. If you set a completion alert for a baseline, the alert is reported when all nodes of the baseline are completed.

- **Uncompleted**

You can set alerts for nodes, baselines, or workspaces that are not completed at a certain time point. For example, if you require that a baseline be completed at 10:00, an alert containing a list of uncompleted nodes is reported once a node in the baseline is not completed at the specified time.

- **Error**

An error alert can be set for nodes, baselines, and workspaces. Once a node has an error, an alert containing detailed node error information is sent to the recipient.

- **Uncompleted Cycle**

For the monitoring rules of hourly scheduled nodes, you can separately specify the uncompleted time points in different periods.

- **Overtime**

An overtime alert can be set for nodes, baselines, and workspaces. Once a monitored node of the preset object is not completed within the specified time, an alert is reported.

2.8.7.3. Instructions

2.8.7.3.1. Baseline instances

On the Baseline Instances page, you can view the information about a baseline.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**. The **Operation Center** page appears.
3. On the Operation Center page, choose **Monitor > Baselines** in the left-side navigation pane. On the **Baselines** page, create a baseline. For more information about how to create a baseline, see [Create a baseline](#).

 **Note** If you have created a baseline, skip this step and directly go to the **Baseline Instances** page to perform subsequent operations.

4. In the left-side navigation pane, choose **Monitor > Baseline Instances**. The **Baseline Instances** page appears. On this page, you can search for baseline instances by condition, such as the data timestamp, owner name or ID, event ID, workspace, and baseline name. You can also click **View Details**, **Handle**, and **View Gantt Chart** in the Actions column to perform corresponding operations on a baseline.

 **Note** After creating a baseline, you must enable the baseline so that a baseline instance can be generated.

A baseline can be in the one of the following four states:

- **Normal**: All nodes in the baseline are completed before the alerting time.
- **Alerting**: One or more nodes in the baseline are not completed at the alerting time but the committed completion time has not arrived.
- **Overtime**: One or more nodes in the baseline are not completed at the committed completion time.
- **Others**: All nodes in the baseline are paused or the baseline is not associated with any node.

You can click **View Details**, **Handle**, and **View Gantt Chart** in the Actions column to perform corresponding operations on a baseline.

- **View Details**: Click **View Details** to go to the **Baseline Instance Details** page.

On the **Baseline Instance Details** page, you can view the general information, critical path, baseline instance information, history graph, and relevant events.

 **Note**

- In the preceding figure, the data timestamp is **one day before the system time**.
- When you create a baseline, you can select **By the Day Interval** or **By the Hour Interval**. The **Cycle** parameter appears as the advanced settings of the **Committed Time** parameter only when you select **By the Hour Interval**.

- **Handle**: Click **Handle** to pause the alert generated by the baseline when the baseline is being handled.
- **View Gantt Chart**: Click **View Gantt Chart** to view the critical paths of nodes.

2.8.7.3.2. Baselines

You can create and define baselines on the Baselines page.

Create a baseline

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**. The **Operation Center** page appears.
3. On the Operation Center page, choose **Monitor > Baselines** in the left-side navigation pane.
4. On the **Baselines** page, click **Create Baseline** in the upper-right corner.

 **Note** Currently, only workspace administrators can create baselines.

5. In the **Create Baseline** dialog box that appears, set the parameters and click **OK**.

Parameter	Description
Baseline Name	The name of the baseline.
Workspace	The workspace of the node associated with the baseline.
Owner	The name or ID of the owner.
Recurrence	Specifies whether the baseline detects nodes by day or hour. <ul style="list-style-type: none"> ○ By the Day Interval: Select this option for nodes scheduled by day. ○ By the Hour Interval: Select this option for nodes scheduled by hour.
Nodes	<ul style="list-style-type: none"> ○ Node: the node associated with the baseline. Enter the name or ID of a node, and then click the icon on the right to add the node. You can add multiple nodes. ○ Workflow: the workflow associated with the baseline. Enter the name or ID of a workflow, and then click the icon on the right to add the workflow. We recommend that you only add the last node of a workflow instead of all nodes.
Priority	The priority of the baseline. A baseline with a higher priority is scheduled first. Currently, the only available priority value is 1.

Parameter	Description
Estimated Completion Time	The completion time of the node estimated based on the average running time of the node during previous scheduling. If no historical data is available, the message The completion time cannot be estimated due to a lack of historical data appears.
Committed Time	The time point when a node should be completed. An alert is triggered if the node is not completed until the time point obtained by subtracting the alert margin threshold from the committed completion time.
Margin Threshold	<p>The interval before an alert is triggered. For example, set Committed Time to 3:30 and Margin Threshold to 10 minutes. An alert is triggered if the node is not completed at 3:20. Assume that the average running time of the node is 30 minutes. If the node is not started at 2:50, an alert is triggered.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note The average running time of a node can be calculated based on the data of the last 15 days.</p> </div>

6. After a baseline is created, click **Enable** in the Actions column to enable the baseline.

You can click **View Details**, **Change**, **Enable**, **Disable**, or **Delete** in the Actions column to perform the corresponding operation on a baseline.

- **View Details:** Click **View Details** to view the basic information about the baseline.
- **Change:** Click **Change** to modify the baseline.
- **Enable** or **Disable:** Click **Enable** or **Disable** to enable or disable the baseline. A baseline instance can be generated only when the corresponding baseline is enabled.
- **Delete:** Click **Delete** to delete the baseline.

Add a node to a baseline

By default, all nodes in the production environment are in the default baseline of each workspace. When you create a baseline, you actually move nodes from the default baseline to the baseline that you create.

 **Note** A node must belong to a baseline, so you cannot directly remove nodes from the default baseline. Instead, you can create a baseline to move nodes from the default baseline to the new baseline. When you delete a baseline that you create, you actually move the nodes in the baseline to the default baseline.

To change the baseline of a node, perform one of the following operations:

- On the **Baselines** page, click **Create Baseline** in the upper-right corner. Then, create a baseline by following the instructions in the **Create a baseline** section.
- In the left-side navigation pane, choose **Nodes > Recurring**. On the page that appears, find the node and choose **More > Add to Baseline** in the Actions column.

2.8.7.3.3. Events

On the Events page, you can view all events related to slowdown or errors.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**.

3. On the Operation Center page, choose **Monitor > Events** in the left-side navigation pane. The **Events** page appears.

You can search for events by condition, such as the event owner, time when an event was detected, event status, event type, and name or ID of a node or node instance.

In the search results, each event is displayed in a row and associated with a node that encounters errors. The worst baseline indicates a baseline with the minimum margin among the baselines affected by an event.

- Click **View Details** in the Actions column of an event. You can view the event occurrence time, alert time, clearance time, historical runtime logs of the node, and detailed node logs.

You can assign an alert recipient. After you click **View Alerts**, the alert details page corresponding to the event appears. Affected baselines are all descendant baselines affected by the node associated with the event. You can observe descendant baselines and the impact on these baselines and analyze node logs to identify the causes of the event.

- If you click **Handle**, DataWorks records the event handling operation and pauses the alert for the event when the event is being handled.
- If you click **Ignore**, DataWorks keeps the event ignorance record and permanently stops the alert for the event.

2.8.7.3.4. Alert triggers

This topic describes how to customize alert triggers on the Alert Triggers page.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**.
3. On the Operation Center page, choose **Monitor > Alert Triggers** in the left-side navigation pane. The **Alert Triggers** page appears.
4. On the Alert Triggers page, click **Create Custom Trigger** in the upper-right corner.
5. In the **Create Custom Trigger** dialog box that appears, set relevant parameters.

Parameter	Description
Trigger Name	The name of the custom alert trigger.
Object Type	The granularity of monitored objects. Valid values: Node and Workflow .
Objects	The monitored object. Enter the name or ID of a node or workflow and click the icon on the right to add the object.
Trigger Condition	The condition for triggering an alert. Valid values: Completed , Uncompleted , Error , Uncompleted Cycle , and Overtime .
Maximum Alerts	The maximum number of alert reporting times. If the number of alert reporting times exceeds the preset threshold, no alerts are reported.
Minimum Alert Interval	The minimum time interval at which DataWorks reports an alert.
Quiet Hours	The specified period during which no alerts are reported.
Notification Method	The method of reporting alerts. Valid values: Email and SMS .

Parameter	Description
Recipient	The person who receives alerts. You can set this parameter to the node owner or another recipient.
DingTalk Chatbot	The DingTalk chatbot to receive alerts.

6. Click **OK** to create the alert trigger.

On the **Alert Triggers** page, you can click **View Details** in the Actions column of an alert trigger to view the details of the alert trigger.

2.8.7.3.5. Alert information

You can view all alerts on the Alerts page.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**.
3. On the Operation Center page, choose **Monitor > Alerts** in the left-side navigation pane. The **Alerts** page appears.

You can search for alerts by condition, such as the alert trigger ID or name, recipient, alert time, notification method, and alert trigger type.

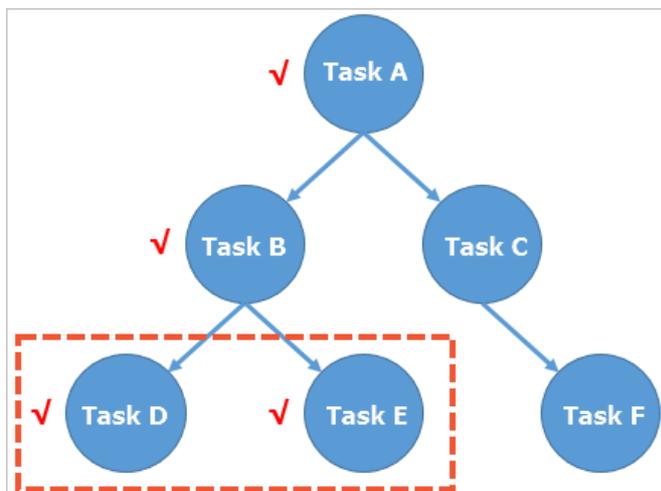
You can also view alert details such as the notification method and status. To view more alert details, find the target alert and click **View Details** in the Actions column.

2.8.7.4. FAQ

This topic describes the FAQ about the Monitor service.

What can I do if I do not need to receive alerts for a node?

After you create and enable a monitoring baseline, the Monitor service monitors all nodes in the baseline and their ancestor nodes. If a node in the baseline or an ancestor node of the baseline affects data generation of the monitored nodes in the baseline, the Monitor service reports an alert to the node owner.



As shown in the preceding figure, assume that DataWorks has only six nodes, and Nodes D and E belong to a monitoring baseline. The Monitor service monitors Nodes D and E and all their ancestor nodes. Namely, the Monitor service will detect any error or slowdown on Node A, B, D, or E. Nodes C and F are not monitored by the Monitor service.

Nodes A and B are ancestor nodes of Nodes D and E and may affect data generation of the monitored nodes in the baseline. When an error or slowdown occurs on Node A or B, the Monitor service reports an alert to the node owner.

If you do not need to receive alerts for a node, use the following methods:

- If the owners of Nodes D and E do not need to receive alerts, contact the baseline owner to remove Nodes D and E from the baseline.
- If the owner of Node A or B does not need to receive alerts, contact the owners of Nodes D and E to delete the dependency of Nodes D and E on Node A or B.

Why is no alert reported for a baseline in the Overtime state?

Baseline monitoring is controlled by the baseline switch and enabled for nodes. If all nodes are running properly, no alert will be triggered even in the Overtime state. This is because all the nodes are running properly and the Monitor service cannot determine which node has an error. Overtime is a baseline state, indicating that a node is still not completed after the committed time.

If the baseline still enters the Overtime state when all nodes are running properly, consider the following reasons:

- The baseline time is not properly set.
- The node dependency is not properly configured.

Can I disable the Monitor service from reporting an alert for a node that slows down?

The Monitor service reports a node slowdown alert only when a node meets both of the following conditions:

- The node is an ancestor node of an important baseline.
- Compared with its historical performance, the node does slow down.

You can view the descendant baseline affected by the node on the **Event Management** page. Then, you can confirm the impact with the party whose monitoring baseline contains descendant nodes of your node.

- If the node slowdown has a minor impact, you can ignore the alert.
- If the node slowdown has a major impact, maintain your node properly.

Why do I fail to receive an alert for an error node?

The Monitor service reports an alert only for specified nodes when an error occurs. An alert is reported for an error node only when the node meets one of the following conditions:

- The node is an ancestor node of a baseline that has been enabled.
- An alert trigger has been customized.

What can I do if I receive an alert at night?

1. [Log on to the DataWorks console.](#)
2. On the **DataStudio** page, click  in the upper-left corner and choose **All Products > Operation Center**.
3. In the left-side navigation pane, choose **Alarm > Event Management**.
4. On the **Event Management** page, disable the event alert. Disable the event alert in one of the following

ways:

- o Handle the event that triggers an alert.
 - a. Find the target event and click **Handle** in the Actions column.
 - b. In the **Handle Event** dialog box, set the **Handling Time** parameter.
 - c. Click **OK**.

 **Note** DataWorks records the event handling operation and pauses alerting for the event when the event is being handled.

- o Ignore the event that triggers an alert.
 - a. Find the target event and click **Ignore** in the Actions column.
 - b. In the **Ignore Event** message, click **OK**.

 **Note** DataWorks records the event ignoring operation and permanently stops alerting for the event.

2.9. Security Center

2.9.1. Overview

Security Center provides flexible permission management features. It allows you to request permissions and handle permission requests on the graphical user interface (GUI), and view and manage permissions. Security Center not only improves data security but also facilitates data permission management.

Log on to the DataWorks console. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Security Center**. The **Security Center** page appears.

Security Center consists of the following modules: **My Permissions**, **Authorizations**, and **Approval Center**.

Currently, Security Center provides the following features:

- **Self-service permission request:** Users can select the required tables to quickly initiate a permission request online. This online request mode is more efficient than the original mode in which users need to contact administrators offline.
- **Permission management:** Administrators can view the users who have permissions on database tables and revoke permissions as required. Users can also revoke unnecessary permissions themselves.
- **Permission request approval:** Before granting permissions to users, administrators approve permission requests initiated by users. This implements a visual and process-based permission management system, and supports reviewing the approval process.

In Security Center, you can view permissions on all the tables in an organization, request and revoke table permissions, and approve or reject permission requests.

Each operation in Security Center applies to all the workspaces of a tenant in standard mode and basic mode.

2.9.2. My Permissions

On the My Permissions page, you can view your table and field permissions in a workspace, and request or revoke table and field permissions.

View table and field permissions

1. Move the pointer over the DataWorks icon in the upper-left corner, and click **Security Center**. In the left-side navigation pane, click **My Permissions**. The **Table** tab appears.

- On this tab, you can select a workspace and specify the environment (for a workspace in standard mode) to view all the tables of the workspace in the specified environment. You can also enter a table name in the search box to search for required tables in fuzzy match mode.

You can view the names and owners of tables in the workspace, view your permissions on the tables, and request or revoke table and field permissions.

Request table and field permissions

- Select the tables and fields on which you want to request permissions.

- Request permissions on a table or some fields in the table

Select the required fields on which you have no permissions in a table and choose **More > Request Permission** in the Actions column.

Alternatively, choose **More > Request Permission** in the Actions column for a table without selecting any fields to request permissions on all the fields in the table.

Note You can request permissions on fields only in a workspace with LabelSecurity enabled. If LabelSecurity is disabled for a workspace, you can request permissions only on tables in this workspace.

- Request permissions on multiple tables and fields

Select all the required tables and fields and click **Request Permission**.

Note You can also click **Request Permission** without selecting any tables or fields, and then select the required tables and fields in the **Table Permission Request** dialog box.

- Set the parameters in the **Table Permission Request** dialog box.

Parameter	Description
Workspace	The name of the workspace, which is automatically entered based on the information you specified on the My Permissions page. You can change the workspace as required.
Environment	The environment of the workspace.
MaxCompute Project	The name of the MaxCompute project.
Grant To	The account for which you request permissions. You can request permissions for the current account or a production account of another workspace you joined.
Reason for Request	The reason why you request permissions.
Objects Requested	The tables on which you request permissions. The tables that you select on the previous page are displayed. You can add tables or delete existing tables as required.

- After the configuration is completed, click **Submit**. If you do not want to request the permissions, click **Cancel**.

Revoke permissions

You can revoke table and field permissions.

- Revoke field permissions

 **Note**

- You can revoke permissions on fields only in a workspace with LabelSecurity enabled.
- To revoke permissions on all the fields in a table, you can directly revoke the permissions on the table.

- Choose **More > Revoke Field Permission** in the **Actions** column for the table on which you want to revoke permissions.
 - In the **Revoke Field Permission** dialog box, select the fields on which you want to revoke permissions.
 - Click **OK**.
- Revoke table permissions
 - Choose **More > Revoke Permission** in the **Actions** column for the table on which you want to revoke permissions.
 - In the **Revoke Permission** dialog box, select the permissions you want to revoke.
 - Click **OK**.

2.9.3. Authorizations

On the Authorizations page, a workspace administrator can view the accounts that have permissions on tables and fields in each workspace, and revoke unnecessary table and field permissions.

You can move the pointer over the DataWorks icon in the upper-left corner, and click **Security Center**. In the left-side navigation pane, click **Authorizations**. On the **Table** tab that appears, you can view and search for tables in workspaces of the current organization.

On the Table tab, you can select a workspace and specify the environment (for a workspace in standard mode) to view all the tables of the workspace in the specified environment. You can also enter a table name in the search box to search for required tables in fuzzy match mode.

View accounts that have permissions on a table

On the **Table** tab of the **Authorizations** page, click the plus sign (+) in front of a table to view all the accounts that have permissions on the table.

Revoke table permissions

Click **Revoke Permission** in the **Actions** column for an account to revoke the permissions of the account on the current table.

View field permissions

Click **View Field Permissions** in the **Actions** column for an account to view the permissions of the account on the fields in the current table.

Revoke field permissions

If LabelSecurity is enabled for the workspace, select fields on the Field Permissions page and click **Revoke Field Permissions** to revoke the permissions on the fields.

2.9.4. Approval Center

On the Approval Center page, you can view your requests and their status, view and handle the requests pending your approval, and view the requests that you have handled.

My Requests

1. Move the pointer over the DataWorks icon in the upper-left corner, and click **Security Center**. In the left-side navigation pane, click **Approval Center**. On the Approval Center page, click the **My Requests** tab.

On this tab, you can view the information about each of your requests, including **Object Type**, **Workspace**, **Status**, **MaxCompute Project**, **Request Time**, and **Table**.

 **Note** If a request contains permission requests for tables that belong to different owners, Security Center automatically splits the request into multiple requests by table owner.

2. Click **View** in the Actions column to view the details about a request.

Pending My Approval

1. On the **Approval Center** page, click the **Pending My Approval** tab.

On this tab, you can view the requests pending your approval. If a request is pending your approval, a red dot appears next to Approval Center and Pending My Approval to remind you.

You can view the information about each of requests pending your approval, including **Object Type**, **Grant To**, **Request Time**, **Workspace**, **MaxCompute Project**, and **Table**.

2. Click **Handle** in the Actions column to view the details about a request and handle it on the Request Details page. The request details include the progress and objects requested.
3. Enter your comments and click Approve or Reject as required.

Handled by Me

1. On the **Approval Center** page, click the **Handled by Me** tab.

On this tab, you can view the information about each request that you have handled, including **Object Type**, **Grant To**, **Result**, **Workspace**, **MaxCompute Project**, **Table**, and **Request Time**.

2. Click **View** in the Actions column to view the details about a request. The request details include the progress and objects requested.

2.9.5. FAQ

This topic describes the frequently asked questions (FAQs) about the Security Center service of DataWorks.

- Q: What permissions can I request in Security Center?

A: In Security Center, you can request permissions on tables in DataWorks workspaces in the development environment and production environment.

- Q: What is the relationship between Data Management and Security Center?

A: Security Center is a product that upgrades and replaces the permission and security features in Data Management. You can choose **Security Center > My Permissions** to view the permissions requested or granted by using the `odpscmd grant` command in **Data Management**.

If you want to request other permissions and handle permission requests on the GUI, go to **Security Center** and perform operations as required. The **Data Management** service does not support permission request and approval any more.

- Q: Why cannot I select fields when I request permissions?

A: If LabelSecurity is enabled for a workspace, you can request permissions on fields in this workspace. If LabelSecurity is disabled for a workspace, you can request permissions only on tables in this workspace.

- Q: Who will handle my request?

A: Your request is handled by a workspace administrator or a table owner. After either of them approves or rejects your request, the request is closed.

- Q: Why do I find two requests on the **My Requests** page after I submit only one request?

A: The tables in your request belong to two owners. In this case, Security Center automatically splits your request into two by table owner.

- Q: I request permissions on a field for one month only. Why does the validity period of the permissions become permanent after my request is approved?

A: The security level of this field is zero or not higher than the security level of your account.

- Q: Why do I obtain permissions on some tables and fields on which I have not requested any permissions?

A: The possible causes are as follows:

- An administrator has granted the permissions to you by running commands in the DataWorks console.
- After your request is approved in Security Center, Security Center also grants you the permissions on fields whose security level is zero or not higher than the security level of your account, even though you have not requested the permissions.

- Q: Why does a request disappear from the **Pending My Approval** tab before I handle it?

A: Another workspace administrator or the table owner has approved or rejected the request. The request is closed and no longer needs to be handled.

- Q: What can I do if the message "An error occurred in the MaxCompute project" appears when I specify the workspace and environment?

A: Send the error message and error code to a workspace administrator for troubleshooting.

- Q: Why do I fail to revoke permissions on a field?

A: You can revoke permissions only on the fields whose security level is higher than the security level of your account.

- Q: Why do I fail to request permissions by using my tenant account?

A: By default, a tenant account has all permissions. Therefore, you do not need to request permissions for your tenant account. The tenant account hides unnecessary operations such as permission request. This does not affect the use of the tenant account.

- Q: In **Security Center**, can I view the permission request and approval records of **Data Management**?

A: Security Center and Data Management have not synchronized permission request and approval records yet. You need to go to **Data Management** to view the permission request and approval records of Data Management.

- Q: Can I revoke permissions based on the request records in Security Center?

A: Currently, Security Center is not the only service that provides authorization. To facilitate permission revocation, the Authorizations page in Security Center provides an access control list (ACL) of all users, regardless of the authorization channel. You can revoke any granted permissions without using the request records.

- Q: A permission request submitted in **Data Management** has not been approved yet. Do I need to submit it again in Security Center?

A: Security Center and Data Management have not synchronized permission request and approval records yet. You need to submit the permission request again in Security Center.

- Q: How do I specify the LabelSecurity parameter for fields?

A: You need to go to **Data Map** to set the LabelSecurity parameter for fields.

2.10. Data Quality

2.10.1. Overview

DataWorks provides a Data Quality service for you to control the data quality of disparate connections. In Data Quality, you can check data quality, configure alert notifications, and manage connections.

Relying on DataWorks, Data Quality provides a comprehensive data quality solution that has various features. For example, you can detect data, compare data, monitor data quality, and use intelligent alerting.

Data Quality monitors data in datasets. Currently, it allows you to monitor MaxCompute tables and DataHub topics. When offline MaxCompute data changes, Data Quality checks data and blocks nodes if it detects exceptions. This prevents nodes from being affected. In addition, Data Quality allows you to manage the check result history so that you can analyze and evaluate the data quality.

For streaming data, Data Quality uses DataHub to monitor data streams and sends alert notifications to subscribers if it detects stream discontinuity. You can also set the alert severity such as warning and error alerts, and the alert frequency to minimize repeated alerts.

The following figure shows the monitoring flowchart in Data Quality.

 **Note** Data Quality monitors the quality of data from MaxCompute and DataHub datasets. To use Data Quality features, you need to create tables and write data to the tables.

You can create MaxCompute tables and write data to the tables in the MaxCompute console or in the DataWorks console.

Log on to the DataWorks console. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Quality** to go to the **Data Quality** page.

2.10.2. Features

2.10.2.1. Dashboard

As the homepage of Data Quality, the Dashboard page displays an overview of alerts and blocks for subscribed nodes. You can set filter conditions to view the required alerts and blocks.

Card	Description
My MaxCompute Partition Subscriptions	Displays the number of MaxCompute partitions with alerts or blocks and the number of normal MaxCompute partitions on the current day. You can click this card to go to the Search by Node page for the MaxCompute connection and view alert details.
My DataHub Topic Subscriptions	Displays the number of DataHub topics with alerts and the number of normal DataHub topics on the current day. You can click this card to go to the Search by Node page for the DataHub connection and view alert details.
Current task alarm condition	Displays alerts for MaxCompute and DataHub connections of the current workspace on the current day.
Current task blocking situation	Displays blocks for the MaxCompute connection of the current workspace on the current day.

Card	Description
Task Alarm Situation Trend	Displays the trend chart of alerts for MaxCompute and DataHub connections. You can view the alert trend in the past 7 or 30 days, or a custom time period within the past three months.
Task Blocking Situation Trend Graph	Displays the trend chart of blocks for MaxCompute and DataHub connections. You can view the block trend in the past 7 or 30 days, or a custom time period within the past three months.

2.10.2.2. My Subscriptions

The My Subscriptions page displays all nodes subscribed by your account.

Go to the My Subscriptions page

Currently, Data Quality allows you to monitor MaxCompute tables and DataHub topics. You can select a connection on the **My Subscriptions** page and search for subscribed nodes of the connection.

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Data Quality**.
3. In the left-side navigation pane, click **My Subscriptions**. The **My Subscriptions** page appears.

Subscribed MaxCompute connections

On the **My Subscriptions** page, select **MaxCompute** from the connection drop-down list in the upper-left corner. All the subscribed MaxCompute connections appear.

- You can click a partition expression on the right to go to the **Rules** page.
- You can click **View Check Results** in the Actions column for a partition expression to go to the Search by Node page.
- Data Quality supports the following four notification methods: **Email**, **Email and SMS**, **DingTalk Chatbot**, and **DingTalk Chatbot @ALL**.
- You can click **Cancel Subscription** to unsubscribe from the connection.

Subscribed DataHub connections

On the **My Subscriptions** page, select **DataHub** from the connection drop-down list in the upper-left corner. All the subscribed DataHub connections appear.

- After you click **Alerts** for a topic, the **Alerts** page appears, allowing you to view detailed information about the rule alert.
- You can click **Notification Method** for a topic to change the notification method of the rule alert.
- You can click **Cancel Subscription** in the Actions column for a topic to unsubscribe from the topic.

2.10.2.3. Configure monitoring rules

Data Quality can monitor data in the MaxCompute, DataHub, and E-MapReduce data stores. This topic describes how to configure a rule for monitoring a table or topic.

Go to the Monitoring Rules page

1. Log on to the DataWorks console.
2. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Quality**.

3. On the Data Quality page, click **Monitoring Rules** in the left-side navigation pane. On the Monitoring Rules page, you can select ODPS, Datahub, or EMR from the Engine/Data Source drop-down list.
 - If you select ODPS or EMR, all tables in the current MaxCompute or E-MapReduce data store appear. You can also switch to another data store or enter a keyword in the search box to search for topics or tables.
 - If you select Datahub, all topics and dimension tables in the current DataHub data store appear. You can also switch to another data store or enter a keyword in the search box to search for topics or tables.
4. Find the target table or topic and click **View Monitoring Rules** in the Actions column. The rule configuration page for the table or topic appears.

Data Quality allows you to configure template rules and custom rules for a table or topic.

 **Note** Before you configure a template rule for a table, you must configure a partition filter expression.

Create a template rule

1. Click **View Monitoring Rules** in the Actions column of a table or topic.
2. On the rule configuration page that appears, click the partition filter expression for which you want to configure a template rule. Then, click **Create rules**. In the Create rules right-side pane, the **Template Rules** tab appears.

On the **Template Rules** tab, click **Add Monitoring Rule** or **Quick Create** to create a template rule.

- **Click Add Monitoring Rule.**

Set parameters in the rule configuration section that appears, as described in the following table.

Parameter	Description
Rule Name	The name of the rule.
Rule Type	The type of the rule. Valid values: Rule Type and Soft. <ul style="list-style-type: none"> ■ If you select Rule Type, error alerts are reported and descendant nodes are blocked, whereas warning alerts are reported but descendant nodes are not blocked. ■ If you select Soft, error alerts are reported but descendant nodes are not blocked, whereas warning alerts are not reported and descendant nodes are not blocked.
Field	The fields to be monitored. You can select All Fields in Table or select a field of a numeric type or non-numeric type.
Template	The template to apply to the rule. Data Quality supports 37 rule templates. <div style="background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> <p> Note You can set field-specific rules of the average value, accumulated value, minimum value, and maximum value only for numeric fields.</p> </div>
Comparison Method	The comparison method of the rule. Valid values: Absolute Value , Raise , and Drop .

Parameter	Description
Thresholds	<ul style="list-style-type: none"> You can calculate the fluctuation by using the following formula: $\text{Fluctuation} = (\text{Sample} - \text{Baseline}) / \text{Baseline}$ You can calculate the fluctuation variance only for numeric fields such as BIGINT and DOUBLE fields by using the following formula: $\text{Fluctuation variance} = (\text{Sample} - \text{Baseline}) / \text{Standard deviation}$ <div style="background-color: #e1f5fe; padding: 10px; margin: 10px 0;"> <p>Note The sample and baseline are defined in the following way:</p> <ul style="list-style-type: none"> Sample: the sample value for the current day. For example, if you need to check the fluctuation of table rows on an SQL node in a day, the sample is the number of table rows on the current day. Baseline: the comparison value from the previous N days. Examples: <ul style="list-style-type: none"> If you need to check the fluctuation of table rows on an SQL node in a day, the baseline is the number of table rows on the previous day. If you need to check the fluctuation of the average number of table rows on an SQL node in seven days, the baseline is the average number of table rows in the last seven days. </div> <p>You can set Warning Threshold and Error Threshold to monitor data at different severities:</p> <ul style="list-style-type: none"> If the fluctuation does not exceed the warning threshold, Data Quality determines that data is normal. If the fluctuation exceeds the warning threshold but does not exceed the error threshold, Data Quality reports a warning alert. If the fluctuation exceeds the error threshold, Data Quality reports an error alert. If you do not specify the warning threshold, Data Quality reports error alerts or normal based on the monitoring result. If you do not specify the error threshold, Data Quality reports warning alerts or normal based on the monitoring result. If you specify neither the warning threshold nor the error threshold, Data Quality reports error alerts if it detects anomalies. However, you must specify at least one of the two thresholds. If you specify neither of them, Data Quality applies default values, namely, 10% for the warning threshold and 50% for the error threshold.

o **Click Quick Create.**

Set parameters in the rule configuration section that appears, as described in the following table.

Parameter	Description
Rule Name	The name of the rule.
Field	The fields to be monitored. You can select All Fields in Table or a specific field of a numeric type or non-numeric type.

Parameter	Description
Trigger	The trigger condition of the rule. If you select All Fields in Table for the Field parameter, The number of rows is greater than 0 is selected by default.

3. Click **Batch Create**.

Create a custom rule

If template rules do not meet your requirements for monitoring the data quality, you can create custom rules.

1. Click **View Monitoring Rules** in the Actions column of a table or topic.
2. On the rule configuration page that appears, click the partition filter expression for which you want to configure a custom rule. Then, click **Create rules**. In the Create rules right-side pane, the **Template Rules** tab appears.
3. Click the **Custom Rules** tab. On the **Custom Rules** tab, click **Add Monitoring Rule** or **Quick Create** to create a custom rule.
 - o **Click Add Monitoring Rule.**

Set parameters in the rule configuration section that appears, as described in the following table.

Parameter	Description
Rule Name	The name of the rule.
Field	The fields to be monitored. You can select All Fields in Table , SQL Statement , or a specific field. <ul style="list-style-type: none"> ▪ If you select All Fields in Table or a specific field, you can specify the WHERE clause to customize filter conditions based on business requirements. ▪ If you select SQL Statement, you can customize the SQL logic to set a rule. The return value is the value in a row of a column.
Rule Type	The type of the rule. Valid values: Rule Type and Soft. <ul style="list-style-type: none"> ▪ If you select Rule Type, error alerts are reported and descendant nodes are blocked, whereas warning alerts are reported but descendant nodes are not blocked. ▪ If you select Soft, error alerts are reported but descendant nodes are not blocked, whereas warning alerts are not reported and descendant nodes are not blocked.
Sampling Method	The statistical function. Valid values: count and count/table_count .
Filter	The filter condition of the rule. For example, if you need to query partitions of the table based on a specific data timestamp, you can specify <code>pt=\${yyyymmdd-1}</code> as the filter condition.
Check type	The threshold type of the rule. Valid values: Numeric type and Fluctuation .
Comparison Method	The comparison method of the rule. If you set Check type to Numeric type, the values that are optional for this parameter include Greater Than , Greater Than or Equal To , Equal To , Unequal To , Less Than , and Less Than or Equal To .

Parameter	Description
Verification Method	The verification method of the rule. If you set Check type to Numeric type, you can only set this parameter to Compare with a specified value .
Expected Value	The expected value of the rule.
Description	The description of the rule.

o **Click Quick Create.**

Set parameters in the rule configuration section that appears, as described in the following table.

Parameter	Description
Rule Name	The name of the rule.
Trigger	The type of the rule. You can select only Values Duplicated in Multiple Fields .
Field	The fields to be monitored.

Associate a custom node with Data Quality monitoring rules

Before you associate a custom node with Data Quality monitoring rules, make sure that the custom node is created and committed to the production environment. For more information, see [Create a custom node type](#).

You can use one of the following methods to associate a custom node with Data Quality monitoring rules:

- Associate a custom node with Data Quality monitoring rules on the Data Quality page.
 - i. Log on to the DataWorks console.
 - ii. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Quality**.
 - iii. On the Data Quality page, click **Monitoring Rules** in the left-side navigation pane.
 - iv. Select the target data store from the Engine/Data Source drop-down list, find the target table or topic, and then click **View Monitoring Rules** in the Actions column.
 - v. On the rule configuration page that appears, click the partition filter expression for which monitoring rules are configured.
 - vi. Click **Manage Linked Nodes**.
 - vii. In the **Manage Linked Nodes** dialog box, select the target workspace, enter the ID or name of the custom node, and then click **Create**.
- Associate a custom node with Data Quality monitoring rules on the Operation Center page.
 - i. Log on to the DataWorks console.
 - ii. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**.
 - iii. In the left-side navigation pane, choose **Cycle Task Maintenance > Cycle Task**.
 - iv. Find the target node and choose **More > Configure Data Quality Rules** in the Actions column.
 - v. In the **Configure Data Quality Rules** dialog box, set the **Workspace**, **Table Name**, **Engine type**, **Engine instance**, and **Partition Expression** parameters, and click **Add**.

2.10.2.4. View monitoring results

The Node Query page displays the monitoring results of rules. After monitoring rules are triggered, you can go to the Node Query page to view the monitoring results of the rules.

Go to the Node Query page

1. Log on to the DataWorks console.
2. On the DataStudio page, click  in the upper-left corner and choose **All Products > DataAnalysis**.
3. On the Data Quality page, click **Node Query** in the left-side navigation pane. On the **Node Query** page, you can set parameters, such as the **Engine/Data Source**, **Status**, and **My Subscriptions** parameters, to filter nodes and view the monitoring results.

View the monitoring results of E-MapReduce and MaxCompute tables

GUI element	Description
Engine/Data Source	The name of the compute engine. In this example, select EMR or ODPS .
Engine/Database Instance	The E-MapReduce instance or MaxCompute project where the desired tables reside.
Status	The monitoring result of rules. Pay attention to partitions that trigger alerts or blocks.
Data Timestamp	The data timestamp.
My Subscriptions	Specifies whether to display only monitoring results of tables that you subscribed to.
Run At	The time when rules were triggered.
Table Name	The name of the table whose monitoring results you want to view.
Node	The node that triggered rules.
Details	<p>Click Details in the Actions column of a table. On the page that appears, you can perform the following operations on each rule configured for the table:</p> <ul style="list-style-type: none"> • Click View History Check Results in the Actions column of a rule to view the monitoring result history of the rule. • Enter comments on a rule based on the execution status of the rule. Perform the following steps to enter comments on a rule: <ul style="list-style-type: none"> i. Click Problem Handling in the Actions column of the rule. ii. In the Problem Handling dialog box, set the Handling Method and Comments parameters. iii. Click OK. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Notice You can only use the problem handling feature in DataWorks Enterprise Edition or higher.</p> </div> <ul style="list-style-type: none"> • Click Handling Logs in the Actions column of a rule to view the processing history of the rule.
Rules	<p>Click Rules in the Actions column of a table to go to the rule configuration page for the table. On this page, you can view partition filter expressions and rules configured for the table, and modify the rules as required. For more information, see MaxCompute monitoring.</p>

GUI element	Description
View Log	Click View Log in the Actions column of a table to view the operational logs of rules configured for the table.
View Statistics	Click View Statistics in the Actions column of a table to view rule execution information about the table, including the number of rows and the table size.

View the monitoring results of DataHub topics

GUI element	Description
Engine/Data Source	The name of the compute engine. In this example, select Datahub .
Configure a data source	The name of the DataHub connection.
Status	The monitoring result of rules. Pay attention to topics that trigger alerts or blocks.
Topic	The name of the topic whose monitoring results you want to view.
My Subscriptions	Specifies whether to display only monitoring results of topics that you subscribed to.
Clear	Click Clear to clear the filter conditions you specified.
View Log	Click View Log in the Actions column of a topic to view the operational logs of rules configured for the topic.
Alerts	Click Alerts in the Actions column of a topic. On the Alerts page, you can view details about alerts triggered by the topic. On the Alerts page, you can click Close in the Actions column of an alert. In the message that appears, click OK to disable the alert.

2.10.2.5. Report Template Management

On the **Report Template Management** page, you can create a template of data quality reports. DataWorks can periodically generate and send data quality reports based on the template.

Create a report template

1. Log on to the DataWorks console. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Quality**.
2. On the Data Quality page that appears, choose **Configuration > Report Template Management** in the left-side navigation pane. The Report Template Management page appears.
3. Click **Create Report Template**. On the **Create Report Template** page that appears, set required parameters.

Section	Parameter	Description
	Name	The name of the report template.

Section	Parameter	Description
Basic settings	Sending Cycle	The interval at which reports are sent. Valid values: Every Day , Every Week , Every Month , and Do Not Send . If you set Sending Cycle to Every Week or Every Month, you also need to specify the specific day on which reports are sent.
	Timespan	The number of days before the current day. DataWorks generates reports based on the data of those days. The maximum value of this parameter is 30.
Statistics of Rule Configuration The Statistics of Rule Configuration section displays metrics about rule configuration for offline data and real-time data. You can select metrics based on your needs.	Offline data	The metrics about rule configuration for tables in the workspace. The metrics include Table count , Partition expression count , Count of rule on offline data , and Rule coverage on tables . The Rule coverage on tables metric indicates the ratio of tables for which quality monitoring rules are configured.
	Realtime data	The metrics about rule configuration for topics in the workspace. The metrics include Topic count , Count of rule on realtime data , Count of rule on cut off data , Rule coverage on topic , Count of rule on delayed data , and Count of customized rule . The Rule coverage on topic metric indicates the ratio of topics for which quality monitoring rules are configured.
Statistics of Rule Execution The Statistics of Rule Execution section displays metrics about rule running for offline data and real-time data. You can select metrics based on your needs. Quality reports display the selected metrics in charts.	Offline data	The metrics about rule running for tables in the workspace. The metrics are classified into the following types: About rules , About partitions , and About tables .
	Realtime data	The metrics about rule running for topics in the workspace. The metrics are classified into the following types: About messages , About alarms , and About cut-offs .

Section	Parameter	Description
Manage Subscriptions	Subscription Method	The method used to notify report subscribers of new reports. Currently, DataWorks sends emails to notify report subscribers of new reports.
	Recipient	The recipient of report notifications. You can add multiple recipients.
	Actions	The operations that you can perform on the subscription. You can click Save or Cancel in the Actions column of a subscription to save or cancel the subscription.
	Add Subscription	The button that allows you to add a subscription.

- Click **Save** in the upper-right corner. A template of data quality reports is generated.

Preview a report template

After creating a report template, you can click **Preview** in the upper-right corner of the Create Report Template page to view the display format of reports generated based on this template.

 **Note** If report subscribers view reports through email notifications, they can only view the reports in tables. If they view reports on the Data Quality page, they can view reports in tables or charts.

2.10.2.6. Manage rule templates

In Data Quality, you can manage a set of custom rule templates and use the rule templates to improve the efficiency of rule configuration.

Context

You can create a rule template on the **Rule Templates** and **Monitoring Rules** pages. After the rule template is created, you can manage and use it.

Create a rule template on the Rule Template Management page

- Log on to the DataWorks console.
- On the DataStudio page, click  in the upper-left corner and choose **All Products > Data Quality**.
- On the Data Quality page, choose **Configuration > Rule Templates** in the left-side navigation pane.
- On the Rule Templates page, click  and select **Create Folder**.
- In the **Create Folder** dialog box, set the **Name** and **Location** parameters and click **OK**.
- Right-click the folder name and select **Create Rule Template**. You can also rename or delete a folder.
- In the **Create Rule Template** dialog box, set relevant parameters.

New rule Template
✕

* Template name :

* Field :

* Sampling Method :

Set Flag :

Please enter the pre-set statement of SQL. \r\nNote: Write the contents of the set directly, separated by an English comma between multiple statements, with no need for a bonus sign at the end of the statement.

* Check type :

* Verification Method :

* Custom SQL ? :

* Destination folder ? :

Parameter	Description
Template Name	The name of the rule template.
Field	The fields to be monitored and the statistical function. You can only set the two parameters to Custom SQL .
Sampling Method	
Set Flag	The SET clause of the SQL statement for querying the field to be monitored. <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc; font-size: small;"> ? Note Separate multiple statements with commas (,). You do not need to add a semicolon (;) at the end of each statement. </div>
Check type	The threshold type of the rule. Valid values: Numeric type and Fluctuation .

Parameter	Description
Verification Method	<p>The verification method of the rule template. The verification methods that can be selected vary with the threshold type.</p> <ul style="list-style-type: none"> If you set the Check type parameter to Numeric type, you can only set this parameter to Compare with a specified value. If you set the Check type parameter to Fluctuation, the values that are optional for this parameter include Compare the current value with the average value of the last 7 days, Compare the current value with the average value of the last 30 days, Compare the current value with the value 1 day before, Compare the current value with the value 7 days before, Compare the current value with the value 30 days before, The variance between the current value and the value 7 days before, The variance between the current value and the value 30 days before, Compare with the value 1, 7, and 30 days before and Compare with the value of the previous cycle.
Custom SQL	The custom SQL statement. You can use <code>\${tableName}</code> as the table name.
Location	The name of the folder to which you want to store the custom rule template.

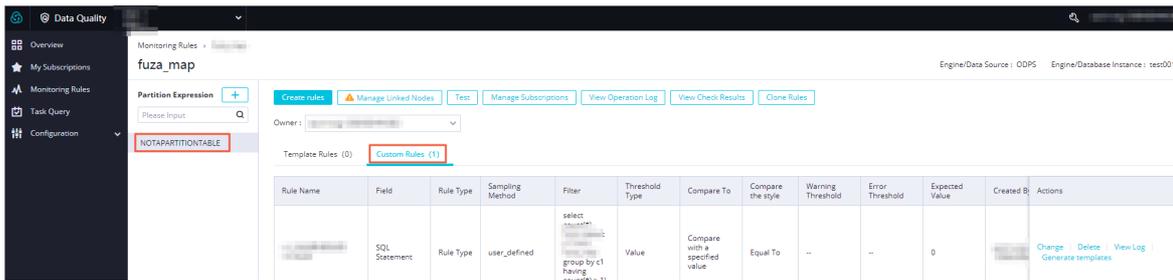
8. Click **OK**.

Create a rule template on the Monitoring Rules page

- Go to the **Data Quality** page.
- On the **Data Quality** page, click **Monitoring Rules** in the left-side navigation pane.
- On the **Monitoring Rules** page, select the compute engine or data store, find the target table or topic, and then click **View Monitoring Rules** in the **Actions** column.

Note This topic uses a MaxCompute table as an example.

- Click a partition filter expression and then click the **Custom Rules** tab.



Note For more information about how to create custom rules, see [Custom rules](#).

- On the **Custom Rules** tab, find the target custom rule and click **Generate Template** in the **Actions** column.
- In the **Create Template** dialog box, set relevant parameters.

New rule Template
✕

* Template name :

* Field :

* Sampling Method :

Set Flag :

Please enter the pre-set statement of SQL. \r\nNote: Write the contents of the set directly, separated by an English comma between multiple statements, with no need for a bonus sign at the end of the statement.

* Check type :

* Verification Method :

* Custom SQL ? :

* Destination folder ? :

Parameter	Description
Template Name	The name of the rule template.
Field	The fields to be monitored and the statistical function. You can only set the two parameters to Custom SQL .
Sampling Method	
Set Flag	The SET clause of the SQL statement for querying the field to be monitored. <div style="background-color: #e6f2ff; padding: 5px; margin-top: 5px;"> ? Note Separate multiple statements with commas (,). You do not need to add a semicolon (;) at the end of each statement. </div>
Check type	The threshold type of the rule. Valid values: Numeric type and Fluctuation .

Parameter	Description
Verification Method	<p>The verification method of the rule template. The verification methods that can be selected vary with the threshold type.</p> <ul style="list-style-type: none"> If you set the Check type parameter to Numeric type, you can only set this parameter to Compare with a specified value. If you set the Check type parameter to Fluctuation, the values that are optional for this parameter include Compare the current value with the average value of the last 7 days, Compare the current value with the average value of the last 30 days, Compare the current value with the value 1 day before, Compare the current value with the value 7 days before, Compare the current value with the value 30 days before, The variance between the current value and the value 7 days before, The variance between the current value and the value 30 days before, Compare with the value 1, 7, and 30 days before and Compare with the value of the previous cycle.
Custom SQL	The custom SQL statement. You can use <code>\$(tableName)</code> as the table name.
Location	The name of the folder to which you want to store the custom rule template.

- Click **OK**.
- In the left-side navigation pane, choose **Configuration > Rule Templates** to view the created rule template.

Manage an existing rule template

On the Rule Templates page, you can click the name of a rule template to go to the template details page. On this page, you can view, edit, delete, or copy the rule template.



Action	Description
View	<p>You can view the parameter configuration, the rules that use the rule template, and logs of the rule template:</p> <ul style="list-style-type: none"> The Application List tab displays the rules that use the rule template. The View Log tab displays the logs of operations performed on the rule template, including the user who performed each operation, the time when each operation was performed, and the operation details.
Edit	Click Edit in the upper-right corner. In the Edit Rule Template dialog box, modify the required parameters, and click OK .
Delete	Click Delete in the upper-right corner. In the Delete Template message, click OK .
Copy	Click Copy in the upper-right corner. In the Copy Rule Template dialog box, set the Template Name and Location parameters and click OK .

Use a rule template

When you create a monitoring rule, you can select a custom rule template to create the rule based on the rule template.

1. Go to the **Data Quality** page.
2. On the Data Quality page, click **Monitoring Rules** in the left-side navigation pane.
3. On the Monitoring Rules page, select the compute engine or data store, find the target table or topic, and then click **View Monitoring Rules** in the Actions column.

 **Note** This topic uses a MaxCompute table as an example.

4. Click a partition filter expression and click the **Custom Rules** tab.
5. On the **Template Rules** tab of the **Create rules** right-side pane, click **Add Monitoring Rule**.
6. Set the parameters for the rule. Specifically, set the **Rule Source** parameter to **Rule Templates** and select a rule template. For more information about the parameter description, see [Rules](#).

Create rules

Template rules Custom rules

Add Monitoring Rule Quick add

* Rule Name : Enter a rule name. Delete

* Rule Type : Rule Type Soft

* Rule source : Rule Template Library

* Field : SQL Statement(user_defined)

* Template :

* Sampling Method : Custom SQL

Set Flag : Please enter the pre-set statement of SQL. \n\nNote: Write the contents of the set directly, separated by an English comma between multiple statements, with no need for a bonus sign at the end of the statement.

* Check type :

* Verification Method : Compare with a specified value

* Custom SQL : tableName

* Comparison Method : Greater Than

* Expected Value : 0

Description :

Batch add Cancel

7. Click **Batch Create**.

2.10.3. User guide

2.10.3.1. Configure monitoring rules for MaxCompute

The Monitoring Rules page is the most important part of Data Quality, where you can configure rules to monitor data in E-MapReduce, MaxCompute, and DataHub. This topic describes how to configure monitoring rules for MaxCompute.

Add a MaxCompute connection

1. Log on to the DataWorks console.
2. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration**.
3. On the Data Integration page, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
4. Click **Add Connection** in the upper-right corner to add a MaxCompute connection.

Select the MaxCompute connection

1. On the current page, click  in the upper-left corner and choose **All Products > Data Quality**.
2. On the Data Quality page, click **Monitoring Rules** in the left-side navigation pane.
3. Select **ODPS** from the **Engine/Data Source** drop-down list to display all tables in the MaxCompute data store. You can search for a table by table name. Fuzzy search based on the initial letters of a table name is supported.
4. Find the target table and click **View Monitoring Rules** in the Actions column.

Configure a partition filter expression

In Data Quality, you must configure rules based on a partition filter expression:

- To configure rules for a non-partitioned table, you can specify **NOTAPARTITIONABLE** as the partition filter expression.
- To configure rules for a partitioned table, you can specify a data timestamp expression, such as `#{yyyymmdd}`, or a regular expression as the partition filter expression.

On the rule configuration page of a table, click **+** next to **Partition Expression** to add a partition filter expression.

You can create a partition filter expression or select a recommended partition filter expression.

- Create a partition filter expression

In the **Add Partition** dialog box, enter a partition filter expression that conforms to the syntax as required. For a non-partitioned table, select **NOTAPARTITIONABLE** from the recommended partition filter expressions.

- For a table with only one partition, follow the format: `Partition key=Partition value`. The partition value can be either a constant or a system parameter. You must configure partition expressions by using the last partition.
- For a table with multiple partitions, follow the format: `Partition key 1\=Partition value/Partition key 2=Partition value/Partition key N=Partition value`. Each partition value can be either a constant or a system parameter. You must use brackets [] to indicate a parameter, such as `#{yyyymmdd-N}`.

The data timestamp configured in a partition filter expression also determines the recurrence of the partition filter expression. For example, if the data timestamp is the date of five days ago, the partition filter expression is triggered every five days. The following table describes supported partition filter expressions.

Partition filter expression	Description
<code>dt=#{yyyymmdd-N}</code>	Indicates N days before.
<code>dt=#{yyyymm01-1}</code>	Indicates the first day of each month.

Partition filter expression	Description
dt=\${yyyyymm01-Nm}	Indicates the first day of the month that is N months before the current month.
dt=\${yyyymmld-1}	Indicates the last day of each month.
dt=\${yyyymmld-1m}	Indicates the last day of the month that is N months before the current month.
dt=\${hh24miss-1/24}	Indicates one hour before the hour specified by the data timestamp.
dt=\${hh24miss-30/24/60}	Indicates half an hour before the hour specified by the data timestamp.
\${yyyymmdd}	Indicates the data timestamp.
\${yyyymmdd-1}	Indicates one day before the data timestamp of the current instance.
\${yyyymmddhh24miss}	Indicates the data timestamp of the current instance. Follow the <code>yyyymmddhh24miss</code> format by understanding the following format description: <ul style="list-style-type: none"> ◦ yyyy indicates a four-digit year. ◦ mm indicates a two-digit month. ◦ dd indicates a two-digit day. ◦ hh24 indicates a two-digit hour (24-hour clock). ◦ mi indicates two-digit minutes. ◦ ss indicates two-digit seconds.
NOTAPARTITIONTABLE	Indicates the partition filter expression of a non-partitioned table.

- Select a recommended partition filter expression

This section uses the `dt partition` as an example to describe how to select a recommended partition filter expression. We recommend that you specify a regular expression as the partition filter expression for a dynamic partitioned table.

- i. In the **Add Partition** dialog box, click the **Partition Expression** field. A drop-down list appears to show you the partition filter expressions recommended by Data Quality.
 - Select a recommended partition filter expression if it meets your expectation.
 - Specify a custom partition filter expression if no recommended partition filter expressions meet your expectation.
- ii. After you enter a partition expression, click **Verify**. Data Quality uses the current time, that is, the data timestamp, to calculate data and verify the partition filter expression.
- iii. Click **OK**.

If you need to delete a partition filter expression, move the pointer over the partition filter expression and click the **Delete** icon to delete the partition filter expression. When you delete a partition filter expression, all rules configured based on the partition filter expression are also deleted.

Link a partition filter expression to a node

To monitor the quality of data involved in a node, you need to link a partition filter expression to the node.

- The Manage Linked Nodes dialog box lists all committed nodes. Data Quality allows you to link a partition filter expression to a node in another workspace.
- Before you link a partition filter expression to a node in another workspace, make sure that you are an administrator, a developer, or an administration expert in the two workspaces.

You can link a partition filter expression to one or more nodes. After nodes are linked, Data Quality can automatically monitor linked nodes.

 **Note** Data Quality allows you to flexibly link a partition filter expression to a node. You can select a node that is not related to your table.

1. On the rule configuration page of a table, click **Manage Linked Nodes**.
2. In **Manage Linked Nodes** dialog box, enter the name of the node that you want to link to the partition filter expression.
3. Click **Create**.

Create a rule

The Monitoring Rules page is the most important part of Data Quality, where you can create rules for your tables.

Data Quality allows you to create template rules and custom rules as needed. If you need to create a template rule or a custom rule, you can click **Add Monitoring Rule** or **Quick Create**. For more information, see [Rules](#).

After rules are configured, you can click **Batch Create** to save all the configured rules for the current partition filter expression.

Creation method	Parameter	Description
Add Monitoring	Rule Name	The name of the rule.
	Rule Type	The type of the rule. Valid values: <ul style="list-style-type: none"> • Rule Type: If a node reaches the error threshold, Data Quality reports an error alert and determines that the node fails. If a node reaches the warning threshold, Data Quality reports a warning alert and determines that the node is successful. • Soft: If a node reaches the error threshold, Data Quality reports an error alert and determines that the node is successful. If a node reaches the warning threshold, Data Quality does not report a warning alert and determines that the node is successful.
	Auto-Generated Threshold	You can use the dynamic threshold feature only in DataWorks Enterprise Edition or higher.
	Rule Source	The source of the rule. Valid values: Built-in Template and Rule Templates .

Rule Creation method	Parameter	Description
	Field	<p>The fields to be monitored. You can select All Fields in Table or a specific field. If you select a field, you can apply the rule to the specified field in the table.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note In this example, select All Fields in Table and set other parameters for the table-specific rule.</p> </div>
	Template	<ul style="list-style-type: none"> • If you set Rule Source to Built-in Template, the built-in table-specific rules appear. • If you set Rule Source to Rule Templates, you must set parameters, such as Sampling Method and Set Flag.
	Comparison Method	The comparison method of the rule. Valid values: Absolute Value , Raise , and Drop .
	Thresholds	The warning threshold and error threshold of the fluctuation. You can adjust the slider to specify thresholds or directly enter thresholds.
	Description	The description of the rule.
Quick Create	Rule Name	The name of the rule.
	Field	The fields to be monitored. You can select All Fields in Table or a specific field. If you select a field, you can apply the rule to the specified field in the table.

Creation method	Parameter	Description
	Trigger	<ul style="list-style-type: none"> The trigger condition of the rule. If you select All Fields in Table for the Field parameter, you can set this parameter to The number of columns is greater than 0 or Table row number dynamic threshold. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 5px 0;"> <p> Notice You can use the dynamic threshold feature only in DataWorks Enterprise Edition or higher.</p> </div> <ul style="list-style-type: none"> If you select a field for the Field parameter, you can select The field value already exists, Null Field, or Unique value dynamic threshold. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 5px 0;"> <p> Notice You can use the dynamic threshold feature only in DataWorks Enterprise Edition or higher.</p> </div>

Test rules

After rules are configured for a partition filter expression, you can test all these rules and view the test results.

 **Note** You can manually run these rules to test their configuration and notification methods. We recommend that you test rules as required.

- On the rule configuration page of a table, click **Test**.
- In the **Test** dialog box, set the **Data Timestamp** parameter.

Parameter	Description
Partition	The partition filter expression for which rules are run. The actual partition key varies with the data timestamp. For a non-partitioned table, use <code>NOPARTITIONTABLE</code> as the partition filter expression.
Data Timestamp	The data timestamp for testing rules. The default value is the current time.

- Click **Test**.
- In the **Test** dialog box, click **The test is complete. Click to view the results** to view the test results on the **Node Query** page.

Manage subscriptions

By default, Data Quality sends notifications to the user who created a partition filter expression. You can add other users so that Data Quality sends notifications to them.

- On the rule configuration page of a table, click **Manage Subscriptions**.
- In the **Manage Subscriptions** dialog box, specify the notification method and notification receiver.

Quality supports the following four methods: **Email**, **Email and SMS**, **DingTalk Chatbot**, and **DingTalk Chatbot @ALL**.

 **Note** Add a DingTalk chatbot and obtain a webhook URL. Then, copy the webhook URL to the Manage Subscriptions dialog box.

3. Click **Save**.

View operational logs

On the rule configuration page of a table, click **View Operation Log**. In the **Operations Logs** right-side pane, you can view the information about each operation, including the user who performed the operation, the time when the operation was performed, and the operation details.

The **Details** column displays the details of each operation performed on the current partition filter expression, including the rule configuration details.

View check results

On the rule configuration page of a table, click **View Check Results** to go to the **Node Query** page. On this page, you can view the check results for all rules under the current partition filter expression.

Clone rules

1. On the rule configuration page of a table, click **Clone Rules**.
2. In **Clone Rules** dialog box, set the **Target Expression** parameter.
3. Select **Clone Subscribers** or **Change Table Names in Custom Rules** as required.
4. Click **Clone**.

2.10.3.2. Configure monitoring rules for DataHub

The Monitoring Rules page is the most important part of Data Quality, where you can configure rules to monitor data in E-MapReduce (EMR), MaxCompute, and DataHub. This topic describes how to configure monitoring rules for DataHub.

Context

DataHub monitoring supports the following features:

- Templates for monitoring stream discontinuity and data latency
- Stream processing features, such as custom Flink SQL, dimension table JOIN, multi-stream JOIN, and window functions

Procedure

1. Add a DataHub data source.
 - i. Log on to the DataWorks console.
 - ii. On the **DataStudio** page, click the icon in the upper-left corner and choose **All Products > Data Aggregation > Data Integration**.
 - iii. On the Data Integration page, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
 - iv. Click **New data source** in the upper-right corner to add a DataHub data source. For more information, see [Configure a DataHub connection](#).
2. Select the DataHub data source.

- i. On the current page, click the icon in the upper-left corner and choose **All Products > Data Quality**.
- ii. On the page that appears, click **Monitoring Rules** in the left-side navigation pane.
- iii. On the Monitoring Rules page, select **Datahub** from the **Engine/Data Source** drop-down list and select the newly added DataHub data source from the Engine/Database Instance drop-down list. All the topics in the selected DataHub data source are displayed.

Parameter	Description
Configure Flink/SLS Resources	After you add a data source, click Configure Flink/SLS Resources to configure Realtime Compute and Log Service resources related to the data source.
Topics	<p>The Topics tab lists all the topics in the DataHub data source. You can click the following buttons in the Actions column for a topic:</p> <ul style="list-style-type: none"> ▪ View Monitoring Rules: Click it to create rules for the topic. You can create template rules and custom rules. ▪ Manage Subscriptions: Click it to view and modify subscribers to the topic, and change the notification method. You can use a DingTalk chatbot to receive notifications. The changed notification method takes effect for all subscribers to the topic.
Dimension Tables	<p>When you create custom rules for a topic, you can create and join dimension tables. If the collected data streams lack some fields for a dimension table, you must supplement fields to data streams before data analysis and declare the dimension table in Data Quality.</p> <p>DataHub supports the dimension tables of ApsaraDB for HBase, Lindorm, ApsaraDB RDS, Tablestore, Taobao Distributed Data Layer (TDDL), and MaxCompute.</p> <p>Flink SQL does not design the data definition language (DDL) syntax for dimension tables. You can use the standard CREATE TABLE statement. However, you must add <code>period for system_time</code> to specify the period of a dimension table and declare that the dimension table stores time-varying data.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note When you declare a dimension table, you must specify the primary key. When you join a dimension table with another table, the ON condition must contain an equivalence condition for each primary key of the tables.</p> </div>

- iv. Click the **Topics** tab. Find the topic for which you want to configure monitoring rules and click **View Monitoring Rules** in the Actions column.
3. On the rule configuration page of the topic, click **Create Rule**.
 4. Create a monitoring rule. In Data Quality, you can create template rules and custom rules.
 - o On the Template Rules tab of the Create rules panel, click **Create Template Rule**. Two templates are available: **Data Delay** and **Stream Discontinuity**.

For example, you can select **Data Delay** for the Template Type parameter.

Parameter	Description
-----------	-------------

Parameter	Description
Rule Name	The name of the rule. The name can be a maximum of 255 characters in length.
Field Type	The fields to be monitored. By default, this parameter is set to All Fields in Table.
Template Type	<ul style="list-style-type: none"> Data Delay: monitors the interval between the time when data is generated and the time when data is written to DataHub based on the data timestamp field. If the interval exceeds a specified threshold, an alert is generated. <div style="background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p> Note</p> <ul style="list-style-type: none"> Before you configure a stream discontinuity rule, you must activate Realtime Compute in Flink and create a project. The data timestamp field supports two data types: <code>TIMESTAMP</code> and <code>STRING (yyyy-MM-dd HH:mm:ss)</code>. </div> <ul style="list-style-type: none"> Stream Discontinuity: monitors the period during which no data is written to DataHub. If the period exceeds a specified threshold, an alert is generated.
Alerts Threshold	The maximum number of alerts generated for data latency. Data Quality reports an alert when the number of alerts generated for data latency exceeds this threshold. This parameter is displayed only when you select Data Delay for the Template Type parameter.
Data Timestamp Field	The data timestamp field of the topic for which the rule is created. This field supports two data types: <code>TIMESTAMP</code> and <code>STRING (yyyy-MM-dd HH:mm:ss)</code> . This parameter is displayed only when you select Data Delay for the Template Type parameter.
Alert Frequency	The interval at which alerts are reported. You can set the alert interval to 10 minutes, 30 minutes, 1 hour, or 2 hours.
Warning Threshold	The warning threshold, in seconds. The value must be an integer and less than the error threshold.
Error Threshold	The error threshold, in seconds. The value must be an integer and greater than the warning threshold.

- If template rules do not meet your requirements for monitoring the data quality of DataHub topics, you can create a custom rule. On the Custom Rules tab of the Create rules panel, click **Create Custom Rule**.

 **Note**

- The field in the `SELECT` clause must be a column. Make sure that you can compare the field values with the warning threshold and error threshold.
- The `FROM` clause must include the current topic and all its columns.

Parameter	Description
Rule Name	The name of the rule. The name must be unique in the topic and can be a maximum of 20 characters in length.
Script	<p>The custom SQL script that is used to set a rule. The return value of the SELECT clause must be unique. You can refer to the following sample statements:</p> <ul style="list-style-type: none"> Use a simple SQL statement. <pre>select id as a from zmr_tst02;</pre> Join the topic and a dimension table named test_dim. <pre>select e.id as eid from zmr_test02 as e join test_dim for system_time as of proctime() as w on e.id=w.id</pre> Join the topic and another topic named dp1test_zmr01. <pre>select count(newtab.biz_date) as aa from (select o.* from zmr_test02 as o join dp1test_zmr01 as p on o.id=p.id)newtab group by id.biz_date,biz_date_str,total_price,'timestamp'</pre>
Warning Threshold	The warning threshold, in minutes. The value must be an integer and less than the error threshold.
Error Threshold	The error threshold, in minutes. The value must be an integer and greater than the warning threshold.
Minimum Alert Interval	The minimum interval at which alerts are reported, in minutes.
Description	The description of the rule.

- Click **Batch Create**. After rules are created for the topic, you can perform the following operations:
 - View Log**: Click it to view the operational logs of the rules.
 - Manage Subscriptions**: Click it to view and modify subscribers to the rules, and change the notification method. The changed notification method takes effect for all the subscribers of the rules.

Data Quality supports the following methods: **Email**, **Email and SMS**, **DingTalk Chatbot**, and **DingTalk Chatbot @ALL**.

 **Note** Add a DingTalk chatbot and obtain the webhook address of the chatbot. Then, copy the webhook address to the Manage Subscriptions dialog box.

2.11. Data Map

2.11.1. Overview

Developed based on Data Management, Data Map uses roles to control the permissions for using different features, such as the permissions for creating and previewing data. Data Map helps you build a better enterprise-level knowledge base.

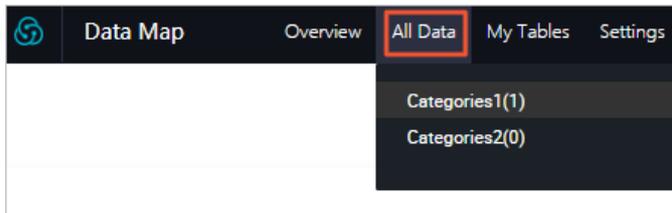
On the homepage of Data Map, you can enter keywords to search for tables by name. You can also click a table in any of the following sections to view the table data: **Recently Viewed Tables**, **Recently Read Tables**, **Most Viewed Tables**, and **Most Read Tables**.

- If you prefer a powerful search engine, go to the homepage to search for data.

Note The homepage appears by default when you access Data Map. To return to the homepage from other pages, click Data Map in the upper-left corner.

- If you need to find tables by project or cluster, click **All Data** in the top navigation bar. On the All Data page, you can view tables on the **MaxCompute** and **E-MapReduce** tabs. On the MaxCompute tab, you can perform the following operations on a table: **Add to Favorites**, **Apply for Permission**, **View Lineage**, and **View DDL Statement**.

If you have added tables to categories, move the pointer over **All Data** in the top navigation bar and select a category. Tables in the category appear. For more information, see [Manage categories and configure workspace permissions](#).



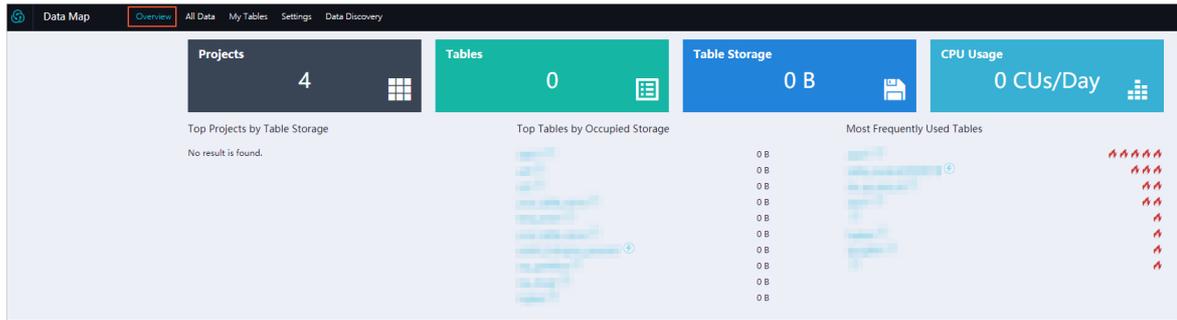
- If you need to view the overall data of the current tenant, click **Overview** in the top navigation bar. For more information, see [View overall data](#).
- If you need to modify tables owned by yourself, click **My Tables** in the top navigation bar. For more information, see [View and manage data and data permissions](#).
- If you are a category administrator or workspace administrator and need to modify the workspace configuration or global categories, click **Settings**. For more information, see [Manage categories and configure workspace permissions](#).

2.11.2. View overall data

This topic describes how to view the overall data of a tenant on the Overview page.

Procedure

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Map(Data Management)**. The homepage of Data Map appears.
3. Click **Overview** in the top navigation bar. The **Overview** page displays offline statistics of the tenant.



Note The Overview page displays statistics of the previous day.

Section	Description
Projects	The total number of projects of the tenant.
Tables	The total number of tables of the tenant.
Table Storage	The total storage occupied by all tables of the tenant.
CPU Usage	The number of compute units (CUs) consumed by the tenant in one day. One CU is equivalent to the computing resources consumed by one fully loaded CPU core in one day.
Top Projects by Table Storage	The top projects that occupy the most storage space under the tenant.
Top Tables by Occupied Storage	<p>The top tables that occupy the most storage space under the tenant. You can click a table name to go to the details page of the table.</p> <p>Note The logical storage space occupied by projects and tables is collected in a T+1 manner. The numbers next to the project and table names indicate the sizes of the occupied logical storage space. The project storage volume includes the table storage volume, storage volumes of resources, data in the recycle bin, and other system files. Therefore, the project storage volume is larger than the table storage volume.</p> <p>The table storage volume is charged based on the logical storage rather than the physical storage.</p>
Most Frequently Used Tables	The most frequently referenced tables of the tenant. You can click a table name to go to the details page of the table.

2.11.3. View and manage data and data permissions

You can view and manage data on the Owned by Me, Managed by Me as Workspace Administrator, Managed by Tenant Account, and My Favorites pages. This topic describes how to view and manage data and data permissions.

Context

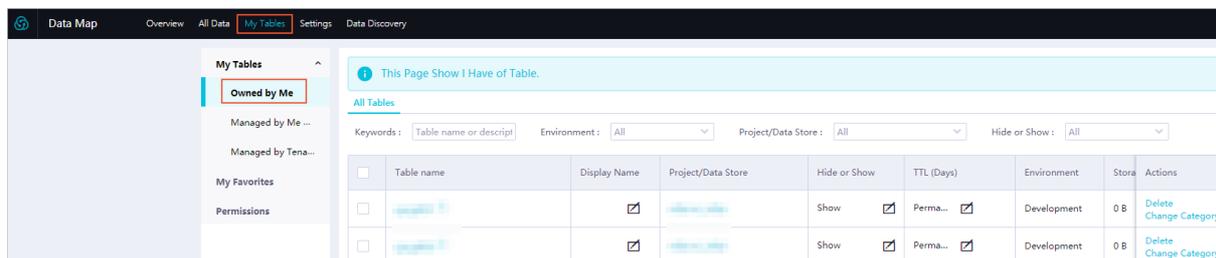
Data Map updates data one day after the data is generated. If you need to query real-time data, we recommend that you use SQL statements.

Go to the My Tables page

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Map(Data Management)**. The homepage of Data Map appears.
3. In the top navigation bar, click **My Tables**. By default, the **My Tables > Owned by Me** page appears.

View and manage data on the Owned by Me page

On the **Owned by Me** page, you can search for data by keyword, environment, project or data store, and visibility. You can also view the details about a table and perform relevant operations on the table.

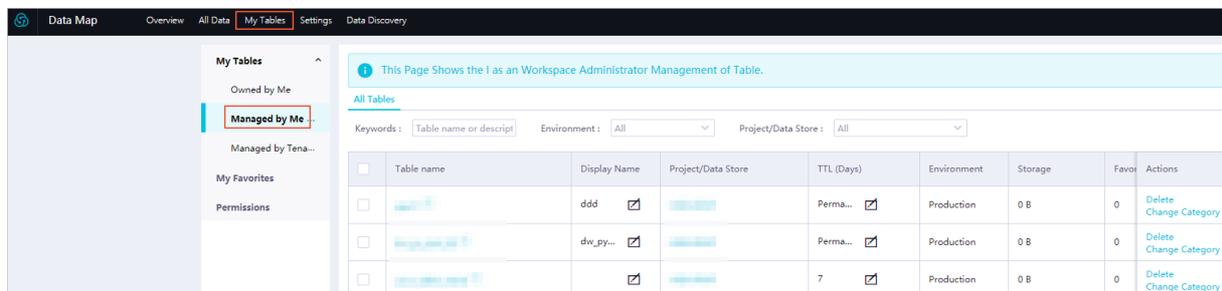


GUI element	Description
Table name	The name of the table. You can click a table name to go to the table details page.
Display Name	The display name of the table. You can click  in the Display Name column of a table to change the display name of the table.
Project/Data Store	The project or data store where the table resides.
Hide or Show	Indicates whether the table is hidden. You can click  in the Hide or Show column of a table to hide or show the table. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note If you hide a table, the Apply for Permissions button does not appear on the details page of the table.</p> </div>
TTL (Days)	The TTL of the table, which is the same as that you set when you create the table.
Environment	The environment to which the table belongs. Valid values: Development and Production .
Storage	The amount of data that is stored in the table.
Favorites	The number of times that users add the table to favorites.
Views in Last 30 Days	The number of times that users view the table in the last 30 days.
Created At	The time when the table was created.

GUI element	Description
Actions	The operations that you can perform on the table. You can click Delete or Change Category in the Actions column of a table to delete the table or change the category of the table.
Edit, Change Owner, Delete, and Change Category	The operations that you can perform on multiple tables at a time. You can select tables and then click Edit , Change Owner , Delete , or Change Category to modify the tables, change the table owners, delete the tables, or change the categories of the tables.

View and manage data on the Managed by Me as Workspace Administrator page

In the left-side navigation pane, click **Managed by Me as Workspace Administrator**. On the page that appears, you can search for data by keyword, project or data store, and environment. You can also view the details about a table and perform relevant operations on the table.

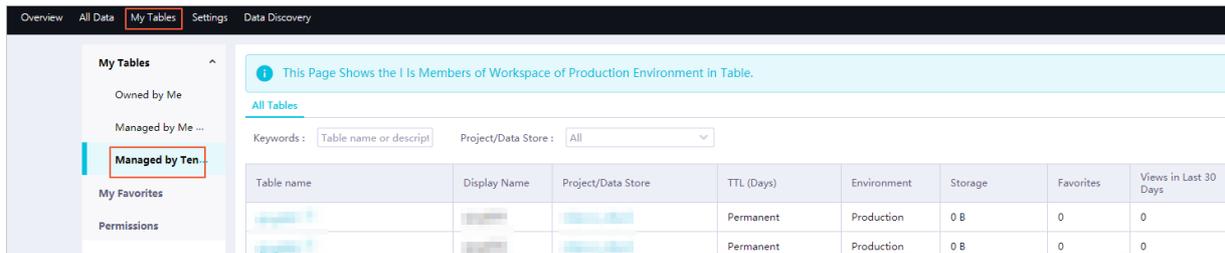


GUI element	Description
Table name	The name of the table. You can click a table name to go to the table details page.
Display Name	The display name of the table. You can click <input checked="" type="checkbox"/> in the Display Name column of a table to change the display name of the table.
Project/Data Store	The project or data store where the table resides.
TTL (Days)	The TTL of the table, which is the same as that you set when you create the table.
Environment	The environment to which the table belongs. Valid values: Development and Production .
Storage	The amount of data that is stored in the table.
Favorites	The number of times that users add the table to favorites.
Views in Last 30 Days	The number of times that users view the table in the last 30 days.
Created At	The time when the table was created.
Actions	The operations that you can perform on the table. You can click Delete or Change Category in the Actions column of a table to delete the table or change the category of the table.

GUI element	Description
Edit, Change Owner, Delete, and Change Category	The operations that you can perform on multiple tables at a time. You can select tables and then click Edit , Change Owner , Delete , or Change Category to modify the tables, change the table owners, delete the tables, or change the categories of the tables.

View data on the Managed by Tenant Account page

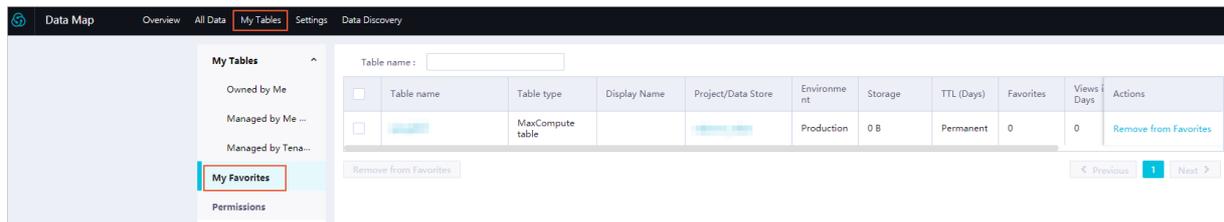
In the left-side navigation pane, click **Managed by Tenant Account**. On the page that appears, you can search for data by keyword and project or data store. You can also view the details about a table and perform relevant operations on the table.



GUI element	Description
Table name	The name of the table. You can click a table name to go to the table details page.
Display Name	The display name of the table. You can click  in the Display Name column of a table to change the display name of the table.
Project/Data Store	The project or data store where the table resides. Tables have different suffixes when they are deployed in different environments. For example, _dev indicates the development environment.
TTL (Days)	The TTL of the table, which is the same as that you set when you create the table.
Environment	The environment to which the table belongs. Valid values: Development and Production .
Storage	The amount of data that is stored in the table.
Favorites	The number of times that users add the table to favorites.
Views in Last 30 Days	The number of times that users view the table in the last 30 days.
Created At	The time when the table was created.

View data on the My Favorites page

In the left-side navigation pane, click **My Favorites**. On the page that appears, you can view the tables that you have added to favorites.



You can click **Remove from Favorites** in the Actions column of a table to remove the table from your favorites.

View and manage data permissions

In the left-side navigation pane, click **Permissions**. On the Permissions page, you can view and manage data permissions.

You can click **Apply function and resource permissions** in the upper-right corner of the **Permissions** page to request permissions. You can also view permission requests on the **To Be Approved**, **Submitted by Me**, and **Handled by Me** tabs.

- **Apply function and resource permissions**

- On the Permissions page, click **Apply function and resource permissions** in the upper-right corner.
- In the **Request Permission** dialog box, set the parameters as required. The following table describes the parameters for requesting permissions.

Request Permission ✕

* **Object Type** : Select an object type.

* **Grant To** : **Current Account** **Specified Account** Specify the account to which you want to grant the permission.

* **Project Name** : Specify a project name.

* **Function Name** : Specify a function name.

Validity Period : If this field is not specified, the permission is permanently valid by default.

* **Reason** : Enter a reason.

Parameter	Description
Object Type	The type of object on which you want to request permissions. Valid values: Function and Resources .

Parameter	Description
Grant To	The account to which the permissions will be granted. Valid values: Current Account and Specified Account . <ul style="list-style-type: none">▪ If you select Current Account, the permissions will be granted to you after the request is approved.▪ If you select Specified Account, you must also specify the Account parameter. The permissions will be granted to the specified account after the request is approved.
Project Name	The name of the MaxCompute project that contains the target function or resource.
Function Name or Resource Name	The full name of the function or resource in the project. If the resource is a file, enter the full name of the file, including the file name extension. Example: my_mr.jar.
Validity Period	The validity period of the permissions, in days. If this parameter is not specified, the permissions are permanently valid. After the validity period expires, the system automatically revokes the permissions.
Reason	The reason why you request the permissions.

- **To Be Approved**

If you are the workspace administrator, you can go to the **To Be Approved** tab. On the tab, you can view and approve the requests for permissions on all objects such as tables, resources, and functions in the workspace.

- **Submitted by Me**

Click the **Submitted by Me** tab on the **Permissions** page.

You can view the permission requests that you have submitted on this tab.

- **Handled by Me**

If you are the workspace administrator, you can go to the **Handled by Me** tab on the **Permissions** page.

On the **Handled by Me** tab, you can view the permission requests that you have handled for all objects such as tables, resources, and functions in the workspace.

2.11.4. Manage categories and configure workspace permissions

This topic describes how to manage categories and configure workspace permissions on the Configuration Management page.

Go to the Configuration Management page

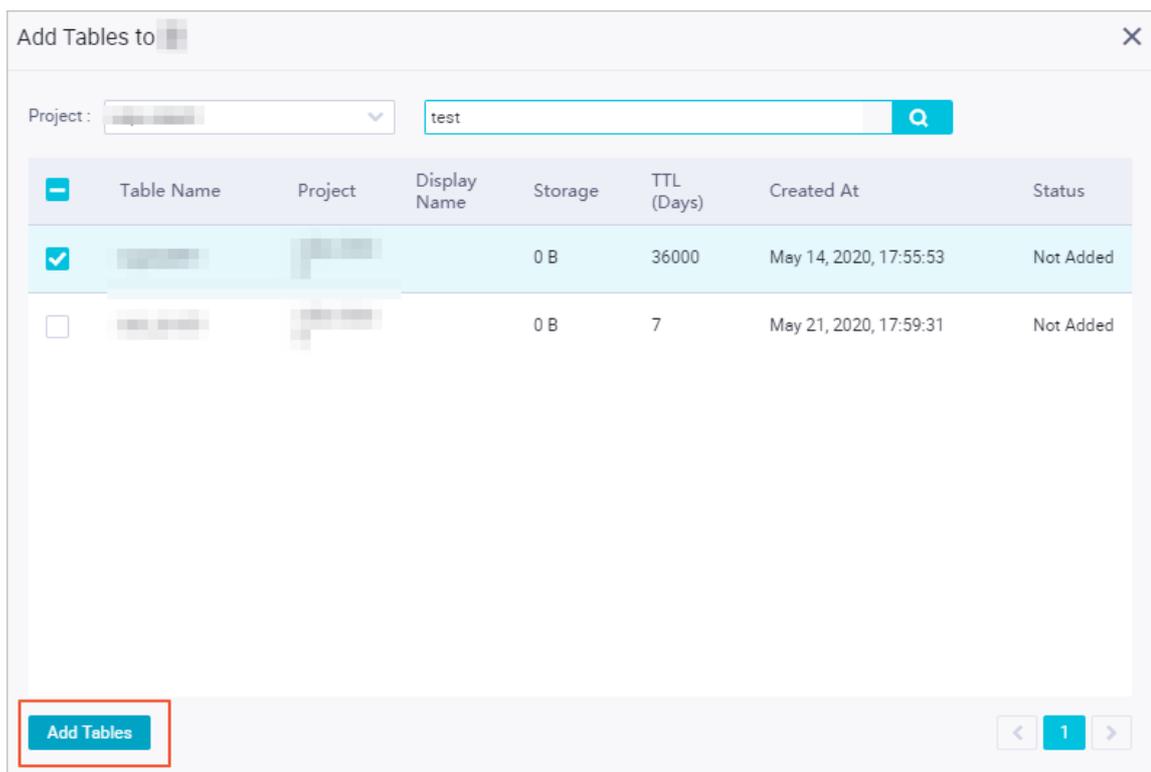
1. Log on to the DataWorks console.
2. Click the  icon in the upper-left corner and choose **All Products > Data Map(Data Management)**.
3. On the Data Map page, click **Configuration Management** in the top navigation bar. The **Manage Categories** tab is automatically opened.

Manage categories

Only tenant administrators have the permissions to create categories and add tables to the categories on the **Manage Categories** page. For more information about how to assign the tenant administrator role to a user, see [Member management](#).

To manage categories as a tenant administrator, perform the following steps:

1. On the **Manage Categories** page, move the pointer over **Categories** and click the **+** icon next to **Categories** to create a level-1 category.
2. Move the pointer over the name of the level-1 category and click the **+** icon next to the level-1 category to create a level-2 category. Use the same method to create more categories. DataWorks allows you to create categories at a maximum of four levels. You can change category names and delete categories.
 - To change the name of a category, move the pointer over the category name, click the **✎** icon, enter a new name, and then press Enter.
 - To delete a category, move the pointer over the category name and click the **🗑** icon. In the **Delete Category** message, click **OK**.
3. Add tables to or remove tables from a category.
 - To add tables to a category, click the name of the category and click **Add Tables** in the upper-right corner. In the Add Tables dialog box, select the tables that you want to add and click **Batch Add**.



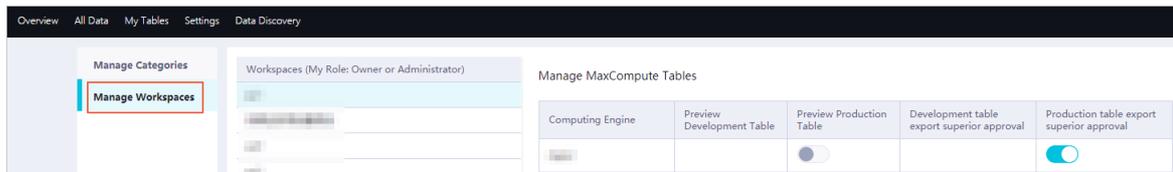
- To remove a table from a category, click the name of the category, find the table that you want to remove, and then click **Remove from Category** in the Actions column. In the message that appears, click **OK**.

Note To remove multiple tables from a category, click the name of the category, select the tables that you want to remove, and then click **Remove from Category** below the table list.

Configure workspace permissions

To configure workspace permissions on the Configuration Management page, perform the following steps:

1. In the left-side navigation pane, click **Manage Workspaces**.
2. Click the name of the required workspace.
3. In the **Manage MaxCompute Tables** section, configure the permissions on the workspace. You can turn on **Preview Tables in Development Environment** and **Preview Tables in Production Environment** as required.



Note If the workspace is in basic mode, you can turn on only **Preview Tables in Production Environment**.

Import and export a category tree

You can import or export a category tree on the **Manage Categories** page.

Note You can export and then import a category tree. You are not allowed to export a category tree, modify it, and then import it.

- Import a category tree
 - i. In the **Manage Categories** section, click **Import**.
 - ii. In the dialog box that appears, select a file that you want to import.
 - iii. Click **Open**.
- Export a category tree

In the **Manage Categories** section, click **Export**.

2.11.5. View table details

2.11.5.1. View the details of a table

This topic describes how to go to the details page of a table and view the details about the table, such as the basic information, output information, and lineage information.

Go to the details page of a table

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Map(Data Management)**. The homepage of Data Map appears.
3. In the top navigation bar, click **All Data**.
4. On the All Data page, click a tab as required.
5. On the tab that appears, click the name of the table you want to view. On the details page that appears, you can view the basic information, business information, permission information, technical information, detailed information (including the fields, partitions, and change history), output information, lineage

information, reference records, and usage notes of the table. You can also preview data in the table.

View basic information

In the **Basic Information** section, you can view the numbers of reads, favorites, and views. You can also check the output nodes, MaxCompute project name, region to which the current workspace belongs, region to which the engine belongs, owner, creation time, time-to-live (TTL), storage capacity, description, and tags of the table, and whether the table is partitioned.

You can perform the following operations in this section:

- View the code of the output node of the table: Click **View Code** next to **Output Node**. On the **Operation Center** page, view the node code.
- View the details about the MaxCompute project: Click the MaxCompute project name. On the page that appears, view the details about the MaxCompute project to which the table belongs.
- Edit the description of the table: Click  next to **Description**, enter a description in the field that appears, and then click .
- Add a tag to or remove a tag from the table: Click  next to **Label**, enter a tag name in the field that appears, and then press **Enter**.

To remove a tag from the table, move the pointer over the tag and click the **Remove** icon.

View business information

In the **Business Information** section, you can view the DataWorks workspace name, environment type, and category of the table.

In this section, you can click the workspace name next to **DataWorks Workspace** to view the details about the workspace to which the table belongs.

View permission information

In the **Permission Information** section, you can view your permissions on the table.

To modify your permissions on the table, perform the following steps:

1. Click **More** in the upper-right corner of the **Permission Information** section.
2. In the **Apply for Data Permissions** dialog box, set the **Grant To**, **Validity Period**, and **Reason** parameters as required.

 **Note** If you do not set the **Validity Period** parameter, the permissions that you request will be permanently valid after your request is approved.

3. Click **Commit**.

View technical information

In the **Technical Information** section, you can view the technical type, last time when the DDL statement was modified, last time when the data was modified, last time when the data was viewed, and compute engine information.

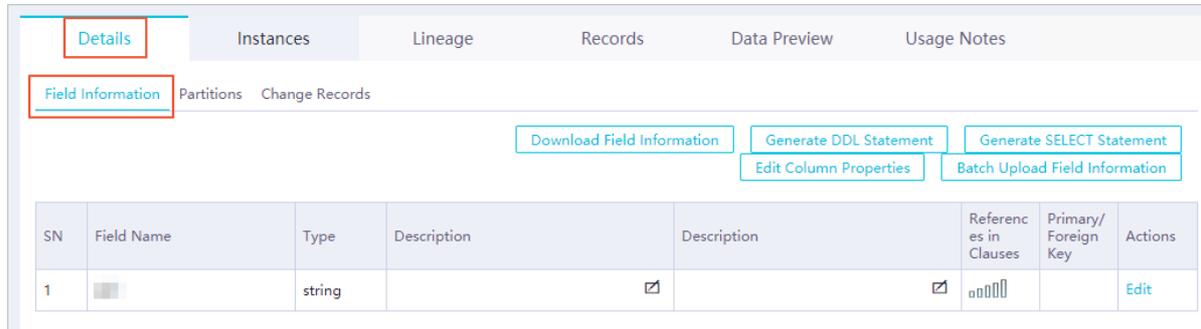
In this section, you can click **Click to View** next to **Compute Engine Information**. In the **Compute Engine Information** dialog box, you can view or copy the information about the compute engine.

View detailed information

The **Details** tab contains the following tabs: **Field Information**, **Partitions**, and **Change Records**.

• **Field Information tab**

On the **Field Information** tab, you can view the name, data type, description, business description, and popularity of fields. You can also check whether a field is a primary key or foreign key.



Button	Description
Edit Column Properties	Click this button, modify the field description and business description, and then click Save or Cancel as needed.
Batch Upload Field Information	Click this button and drag the local file to be uploaded to the Batch Upload Field Information dialog box.
Download Field Information	Click this button to download the field information of the current table.
Generate DDL Statement	Click this button. In the Generate DDL Statement dialog box, view or copy the DDL statement used to create the current table.
Generate SELECT Statement	Click this button. In the Generate SELECT Statement dialog box, view or copy the SELECT statement for querying data in the current table.

• **Partitions tab**

On the **Partitions** tab, you can view the name, number of records, storage capacity, creation time, and last update time of each partition in the current table.

• **Change Records tab**

On the **Change Records** tab, you can view the description, type, granularity, time, and operator of changes performed on the current table.

On this tab, you can also select a change type from the drop-down list in the upper-left corner to filter the table changes.

Change types include **Create Table**, **Modify Table**, **Delete Table**, **Add Partition**, **Delete Partition**, **Change Owner**, and **Modify Lifecycle**.

View output information

If the table data periodically changes with the corresponding node, you can view the change status and data that is continuously updated on the **Instances** tab.

On this tab, you can also click **Code** or **Logs** in the **Actions** column of a node to view the code or logs of the node.

View lineage information

On the **Lineage** tab, you can view the source and destination of data and manage the lineage information with ease.

The **Lineage** tab contains the following tabs: **Table Lineage**, **Field Lineage**, and **Impact Analysis**.

- **Table Lineage** tab: On this tab, you can search for the ancestor and descendant tables of the current table based on the globally unique identifier (GUID).
- **Field Lineage** tab: On this tab, you can select a field from the **Field Name** drop-down list to view the lineage information of the field.
- **Impact Analysis** tab: On this tab, you can query the node that generates a lineage and the full link of the lineage based on information such as the field, node type, table name, project name, and table owner.

You can perform the following operations on the query result:

- Click **Download** to download the query result to your local computer.
- Click **Email**. In the **Notify by Email** dialog box, set the required parameters and click **OK**. In this way, you can configure DataWorks to send email notifications to owners of the descendant tables of the current table.

View reference records

The **Records** tab contains the following tabs: **Foreign Key References** and **Access Statistics**.

- **Foreign Key References** tab: On this tab, you can check the number of users who reference the current table.
- **Access Statistics** tab: On this tab, you can view the reference records in a line chart.

Preview data

On the **Data Preview** tab, you can preview the data of the current table.

 **Notice** Only authorized users can preview tables in the production environment. If you do not have the required permissions, click **Apply Now**.

View usage notes

On the **Instructions** tab, you can edit usage notes, check the historical versions of the usage notes, and view the business description of data.

2.11.5.2. Request table permissions

This topic describes how to request table permissions on the Data Map or Security Center page.

Go to the details page of a table

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Map(Data Management)**. The homepage of Data Map appears.
3. In the top navigation bar, click **All Data**.
4. On the All Data page, click a tab as required.
5. On the tab that appears, click the name of the table on which you want to request permissions.

Request table permissions

1. On the table details page, click **Apply for Permission**.

 **Note** If a table is hidden, the **Apply for Permissions** button does not appear on the details page of the table.

- In the **Apply for Permission** dialog box, set the parameters as required. The following table describes the parameters for requesting table permissions.

Apply for Permission ✕

Table : odps.data1.aaa11

* Grant To : Current Account Specified Account
Specify the account to which you want to grant the permission.

Validity Period : Days
If this field is not specified, the permission is permanently valid by default.

* Reason :
Enter a reason.

Parameter	Description
Table	The table on which you want to request permissions. You cannot change the default value.
Grant To	The type of the account to which the permissions will be granted. Valid values: Current Account and Specified Account .
Username	The username of the account to which the permissions will be granted. <div style="background-color: #e0f2f1; padding: 5px; border: 1px solid #ccc;"> ? Note This parameter is available only when you set the Grant To parameter to Specified Account. </div>
Validity Period	The validity period of the permissions. If you do not set this parameter, the permissions are permanently valid.
Reason	The reason why you request the permissions. Enter a reason for faster approval.

- Click **Commit**.

View the request status

- Click  in the upper-left corner and choose **All Products > Data Map(Data Management)**. The homepage of Data Map appears.
- In the top navigation bar, click **My Tables**.
- On the My Tables page, click **Permissions** in the left-side navigation pane.
- On the page that appears, click the **Submitted by Me** tab.
- Find the target request record and click **View** in the Actions column. The request details appear.

2.11.5.3. Add a table to favorites

This topic describes how to add a table to or remove it from favorites, and view the tables added to favorites.

Procedure

1. Log on to the DataWorks console.
2. Go to the details page of a table and click **Add to Favorites** in the upper-left corner.
3. In the top navigation bar, click **My Tables**.
4. On the **My Tables** page, click **My Favorites** in the left-side navigation pane. On the page that appears, you can view all the tables that you have added to favorites and remove tables from favorites. To remove a table from favorites, find the table in the table list and click **Remove from Favorites** in the Actions column.

2.11.6. Data discovery

2.11.6.1. Collect metadata from an E-MapReduce data source

source

This topic describes how to create a crawler to collect metadata from an E-MapReduce (EMR) data source to DataWorks. You can view the collected metadata on the Data Map page.

Prerequisites

An EMR cluster is bound.

Procedure

1. Go to the **Data Discovery** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data governance > Data Map(Data Management)**. The homepage of Data Map appears.
 - iii. In the top navigation bar, click **Data Discovery**.
2. On the **Obtain Metadata from E-MapReduce** page, click **Create Crawler**.
3. In the **Create Crawler** dialog box, specify **Select Engine Instance** and click **Commit** to enable metadata collection.
4. On the **Obtain Metadata from E-MapReduce** page, find the created crawler and click **Obtain All** in the Actions column. Click **Refresh** in the upper-right corner of the page and verify that the value in the **Running Status** column of the created crawler changes to **The data has been collected**.

 **Note** After full metadata from the EMR data source is collected, the system automatically synchronizes new metadata from the data source.

If you want to delete the created crawler, click **Delete** in the Actions column. In the **Delete Instance** message, click **OK**.

5. View the metadata collected from the EMR data source.
 - i. In the top navigation bar, click **All Data**.
 - ii. Click the **E-MapReduce** tab.
 - iii. On the **E-MapReduce** tab, click the name of the table that stores the collected metadata and view the table details.

2.11.6.2. Collect metadata from a MySQL data source

This topic describes how to create a crawler to collect metadata from a MySQL data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **MySQL**.
3. On the **MySQL Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is MySQL .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source. For more information, see [Configure a MySQL connection](#).
-  **Note** You can select an ApsaraDB RDS for MySQL instance or a MySQL data source that is accessible from the Internet by using a Java Database Connectivity (JDBC) connection string.
- iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **MySQL Metadata Crawler** page, find the created crawler and click **Run** in the **Actions** column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the **MySQL Metadata Crawler** page:

- Click **Details** in the **Actions** column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the **Actions** column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the **Actions** column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the **Actions** column of a running crawler to stop the crawler.

Result

After the crawler is run, you can go to the **MySQL** tab of the **All Data** page to view the table that stores the collected metadata.

2.11.6.3. Collect metadata from a PostgreSQL data source

This topic describes how to create a crawler to collect metadata from a PostgreSQL data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Go to the **Data Discovery** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data governance > Data Map(Data Management)**. The homepage of Data Map appears.
 - iii. In the top navigation bar, click **Data Discovery**.
2. In the left-side navigation pane, click **PostgreSQL**.
3. On the **PostgreSQL Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is PostgreSQL .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source. For more information, see [Configure a PostgreSQL connection](#).
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **PostgreSQL Metadata Crawler** page, find the created crawler and click **Run** in the **Actions** column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the **MySQL Metadata Crawler** page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the **Actions** column of a running crawler to stop the crawler.

2.11.6.4. Collect metadata from an SQL Server data source

This topic describes how to create a crawler to collect metadata from an SQL Server data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **SQL Server**.
3. On the **SQL Server Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is SQL Server .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source. For more information, see [Configure an SQL Server connection](#).
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **SQL Server Metadata Crawler** page, find the created crawler and click **Run** in the Actions column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the **OSS Metadata Crawler** page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.

- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **OK** to delete the crawler.
- Click **Stop** in the **Actions** column of a running crawler to stop the crawler.

2.11.6.5. Collect metadata from an Oracle data source

This topic describes how to create a crawler to collect metadata from an Oracle data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **Oracle**.
3. On the **OracleMetadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is Oracle .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source. For more information, see [Configure an Oracle connection](#).
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **OracleMetadata Crawler** page, find the created crawler and click **Run** in the Actions column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the **OSSMetadata Crawler** page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.

- Click **Delete** in the **Actions** column of a crawler. In the **Confirm** message, click **OK** to delete the crawler.
- Click **Stop** in the **Actions** column of a running crawler to stop the crawler.

2.11.6.6. Collect metadata from an AnalyticDB for PostgreSQL data source

This topic describes how to create a crawler to collect metadata from an AnalyticDB for PostgreSQL data source to DataWorks. After the metadata is collected, you can manage tables in the AnalyticDB for PostgreSQL data source on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **AnalyticDB for PostgreSQL**.
3. On the **AnalyticDB for PostgreSQL Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is AnalyticDB for PostgreSQL .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source.
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **AnalyticDB for PostgreSQL Metadata Crawler** page, find the created crawler and click **Run** in the **Actions** column.

After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

 **Notice** The **Run** button is displayed only in the **Actions** column of a crawler that needs to be manually triggered.

You can also perform the following operations on the AnalyticDB for PostgreSQL Metadata Crawler page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the Actions column of a running crawler to stop the crawler.

2.11.6.7. Collect metadata from an AnalyticDB for MySQL 2.0 data source

This topic describes how to create a crawler to collect metadata from an AnalyticDB for MySQL 2.0 data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **AnalyticDB for MySQL 2.0**.
3. On the **AnalyticDB for MySQL 2.0 Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is AnalyticDB for MySQL 2.0 .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source.
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **AnalyticDB for MySQL 2.0 Metadata Crawler** page, find the created crawler and click **Run** in the Actions column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the MySQLMetadata Crawler page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the Actions column of a running crawler to stop the crawler.

2.11.6.8. Collect metadata from an AnalyticDB for MySQL 3.0 data source

This topic describes how to create a crawler to collect metadata from an AnalyticDB for MySQL 3.0 data source to DataWorks. You can view the collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **AnalyticDB for MySQL 3.0**.
3. On the **AnalyticDB for MySQL 3.0 Metadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is AnalyticDB for MySQL 3.0 .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list. If the required data source does not exist, click **Create Connection** to go to the **Data Source** page in **Workspace Management** and create a data source.
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **AnalyticDB for MySQL 3.0 Metadata Crawler** page, find the created crawler and click **Run** in the Actions column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the MySQLMetadata Crawler page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.
- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the Actions column of a running crawler to stop the crawler.

2.11.6.9. Collect metadata from a Hologres data source

This topic describes how to create a crawler to collect metadata from a Hologres data source to DataWorks. You can view collected metadata on the Data Map page.

Procedure

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **Hologres**.
3. On the **HologresMetadata Crawler** page, click **Create Crawler**.
4. In the **Create Crawler** dialog box, perform the following steps:
 - i. In the **Basic Information** step, configure the basic parameters as required.

Parameter	Description
Crawler Name	Required. The name of the crawler. You must specify a unique name.
Crawler Description	The description of the crawler.
Workspace	The workspace of the data source from which you want to collect metadata.
Connection Type	The type of the data source from which you want to collect metadata. The default value is Hologres .

- ii. Click **Next**.
 - iii. In the **Select Collection Object** step, select a data source from the **Connection** drop-down list.
 - iv. Click **Test Crawler Connectivity**.
 - v. When the message **The connectivity test has been passed.** appears, click **Next**.
 - vi. In the **Configure Execution Plan** step, specify an execution plan. Valid values of the **Execution Plan** parameter: **On-demand Execution**, **Monthly**, **Weekly**, **Daily**, and **Hourly**.
 - vii. Click **Next**.
 - viii. In the **Confirm Information** step, verify that the configuration of the crawler is correct and click **OK**.
5. On the **HologresMetadata Crawler** page, find the created crawler and click **Run** in the Actions column. After the crawler is run, click the number in the **Updated Tables in Last Run** or **Added Tables in Last Run** column to view the details about the updated or added tables.

You can also perform the following operations on the MySQLMetadata Crawler page:

- Click **Details** in the Actions column of a crawler. In the **Crawler Details** dialog box, view the detailed information about the crawler.

- Click **Edit** in the Actions column of a crawler. In the **Edit Crawler** dialog box, modify the configuration of the crawler.
- Click **Delete** in the Actions column of a crawler. In the **Confirm** message, click **Ok** to delete the crawler.
- Click **Stop** in the Actions column of a running crawler to stop the crawler.

2.12. Data Asset Management

2.12.1. Go to the Data Asset Management page

Data Asset Management provides you with an overview of your data assets. Data Asset Management requires that data be synchronized by using Data Integration and processed by using DataStudio before you manage your tables and APIs stored in your business system and DataWorks.

Context

Data Asset Management controls the permissions of users independently. You must grant the permissions on the Project Management page because Data Asset Management is a tenant-level feature.

Data Asset Management allows you to view the metadata collected in Data Map. You can also perform basic management operations on the metadata. For example, you can change the business classes and add business descriptions for metadata tables.

Procedure

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Project Management**. The Project Management page appears.
3. In the left-side navigation pane, click **Member Management**. On the Member Management page, you can assign the following roles to members: **Data Asset Management-Asset Manager**, **Data Asset Management-Class Manager**, and **Data Asset Management-Home Visitor**.
4. Click  in the upper-left corner and choose **All Products > Data Asset Management** to manage data assets in Data Asset Management. The following table describes the permissions of each role.

Role	Permission
Data Asset Management-Asset Manager	In the top navigation bar of the Data Asset Management page, click the Assets tab. On the Assets tab, you can manage data assets, such as adding business units and classes. You can also add data assets to a class.
Data Asset Management-Class Manager	In the top navigation bar of the Data Asset Management page, click the Classes tab. On the Classes tab, you can view the data assets of each class.
Data Asset Management-Home Visitor	In the top navigation bar of the Data Asset Management page, click the Home tab. On the Home tab, you can view the statistics of data assets of different classes and business units.

2.12.2. Asset manager

Asset managers can view the information about data assets.

Procedure

1. Log on to the DataWorks console.

2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Asset Management**.
3. On the **Home** page that appears, enter keywords in the search box and click **Search**.
4. On the search results page that appears, click the **Tables**, **File**, or **API** tab to view details and apply for permissions.

You can click the **Classes** tab in the top navigation bar to filter data assets by class.

2.12.3. Asset user

Asset users can access Data Asset Management to perform operations such as searching for assets, applying for permissions, and using assets.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Asset Management**.
3. On the **Home** page that appears, enter keywords in the search box and click **Search**.
4. On the search results page that appears, click the **Tables**, **File**, or **API** tab to view details and apply for permissions.

You can click the **Classes** tab in the top navigation bar to filter data assets by class.

2.12.4. Asset administrator

Asset administrators can manage assets and authorizations in Data Asset Management.

 **Note** You can submit a ticket to apply for the asset administrator role.

An administrator can grant the administrator role to common users. An administrator can perform any operations in Data Asset Management, and no approval is required.

Go to the Assets tab

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Asset Management**.
3. In the top navigation bar, click the **Assets** tab.

Under the **Assets** tab, you can click **Data Management**, **Classes**, or **Business Management** in the left-side navigation pane to manage your data assets accordingly.

Manage data

In the left-side navigation pane, you can click **Data Management** and then **Tables**, **Files**, or **APIs** to manage tables, files, or APIs.

• Manage tables

In the left-side navigation pane, choose **Data Management > Tables** to go to the **Tables** page.

On the **Tables** page, you can view, edit, publish, or delete a table.

- Click the name of a table to view table details.
- Click **Edit** in the Actions column of a table. In the Edit dialog box that appears, you can edit the configuration of the table.

- Move the pointer over **Publish** in the Actions column of a table. In the dialog box that appears, click **Publish**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You can search for a published table in Data Asset Management.

- Move the pointer over **Delete** in the Actions column of a table. In the dialog box that appears, click **Delete**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You cannot search for a deleted table in Data Asset Management.

• Manage files

In the left-side navigation pane, choose **Data Management > Files** to go to the **Files** page.

On the **File Management** page that appears, you can upload a file. Then, you can view, edit, download, or delete the file.

- In the upper-right corner, click **Upload File**. In the **Upload File** dialog box that appears, click **Add File**. In the Add dialog box that appears, select the file to be uploaded and click **Open**. Alternatively, you can drag and drop a file to the **Upload File** dialog box. Then, click **Next**.

Note

- To upload a file, make sure that the size of the file does not exceed 50 MB.
- You can only upload a file with one of the following file name extensions:
3DX, 7Z, A3D, ATX, AVI, BMP, SV, DBF, DOC, DOCX, DWG, EPS, ESP, FREELIST, GDB, GDBINDEXES, GDBTABLE, GDBTABLX, GIF, GZ, HTM, HTML, IVE, JPEG, JPG, LOCK, LSP, LST, MP3, MP4, MPJ, OSG, OSGB, PDF, PNG, PPT, PPTX, PRJ, PSD, RAR, S3C, SBN, SBX, SCP, SHP, SPX, TFW, TIF, TIFF, TTF, TXT, WAV, WL, WP, WT, XLS, XLSX, ZIP, XML, SHX, and SKP

- Click the name of a file to view file details.
- Click **Edit** in the Actions column of a file. In the Edit dialog box that appears, you can edit the configuration of the file.
- Move the pointer over **Publish** in the Actions column of a file. In the dialog box that appears, click **Publish**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You can search for a published file in Data Asset Management.

- Move the pointer over **Delete** in the Actions column of a file. In the dialog box that appears, click **Delete**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You cannot search for a deleted file in Data Asset Management.

- Click **Download** in the Actions column of a file to download the file.

 **Note** Before downloading a file, apply for the download permission.

• Manage APIs

In the left-side navigation pane, choose **Data Management > APIs** to go to the **APIs** page.

On the **APIs** page, you can edit, publish, or delete an API.

- Click **Edit** in the Actions column of an API. In the **Edit** dialog box that appears, you can edit the configuration of the API. Then, click **Submit**.
- Move the pointer over **Publish** in the Actions column of an API. In the dialog box that appears, click **Publish**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You can search for a published API in Data Asset Management.

- Move the pointer over **Delete** in the Actions column of an API. In the dialog box that appears, click **Delete**. After the request is submitted, click the **Permissions** tab in the top-navigation bar. In the left-side navigation pane, click **Submitted by Me** to view the request you submitted.

 **Note** You cannot search for a deleted API in Data Asset Management.

Manage classes

1. In the left-side navigation pane, click **Classes** to go to the **Classes** page.

On the **Classes** page, you can import or export a class.

2. Click the  icon. In the **Add Class** dialog box that appears, set relevant parameters and click **OK** to add a level-1 class

Parameter	Description
Name	The name of the class, which can be up to 128 characters in length.
Code	The code of the class. This parameter cannot be left empty.
Description	The description of the class.
Confidential	Specifies whether the class is confidential. Valid values: Yes and No .
Share	Specifies whether to share the class. Valid values: Yes , Conditional , and No .

To create a subclass under a class, click the  icon next to the class.

3. Click a class. On the page that appears, click the **Tables** tab.
4. Click **Add Table**. In the **Add Table** dialog box that appears, select the tables to be added to the class and click **OK**.

You can add files and APIs to a class in the same way.

To change the class of a table, click **Modify Class** in the Actions column. In the **Change Class** dialog box that appears, change the class as needed.

Manage business

In the left-side navigation pane, you can click **Business Management** and then **Business Units**, **Business Systems**, or **Connections** to manage business units, business systems, or connections.

-  **Note** Connections belong to a business system, and business systems belong to a business unit.
- A business system with connections cannot be deleted.
 - A business unit with business systems cannot be deleted.

• Manage business units

In the left-side navigation pane, choose **Business Management > Business Units** to go to the **Business Units** page.

Click the  icon. In the **Add Business Unit** dialog box that appears, set relevant parameters and click **OK** to add a business unit.

Parameter	Description
Name	The name of the business unit. This parameter cannot be left empty.
Code	The code of the business unit. By default, the code cannot be modified.
Description	The description of the business unit.
Confidential	Specifies whether the business unit is confidential. Valid values: Yes and No .
Share	Specifies whether to share the business unit. Valid values: Yes , Conditional , and No .
Business system included	Select the business systems to be added to the business unit and click the > icon.

To create a sub-business unit under a business unit, click the  icon next to the business unit.

• Manage business systems

In the left-side navigation pane, choose **Business Management > Business Systems** to go to the **Business Systems** page.

On the **Business Systems** page, you can add a business system. Then, you can view, edit, or delete the business system.

- Click **Add Business System**. In the **Basic information** dialog box that appears, set relevant parameters and click **Submit**.
- Click **View** in the **Actions** column of a business system to view its details.
- Click **Edit** in the **Actions** column of a business system. In the **Business System Properties** dialog box that appears, you can edit the configuration of the business system.
- Click **Delete** in the **Actions** column of a business system. In the **Delete business system** dialog box that appears, click **OK** to delete the business system.

• Manage connections

In the left-side navigation pane, choose **Business Management > Connections** to go to the **Connections** page.

On the **Connections** page, you can view the information of connections. The connection information includes the connection name, the number of tables, the owner, the business system to which the connection belongs, the data type, and the update time. You can also edit the configuration of a connection.

2.12.5. Manage authorizations

Under the **Permissions** tab, you can view permissions in different states on the **Submitted by Me**, **To Be Handled**, **Handled by Me**, and **My Permissions** pages respectively.

Go to the Permissions tab

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Asset Management**.
3. In the top navigation bar, click the **Permissions** tab.

Under the **Permissions** tab, you can view permissions in different states on the **Submitted by Me**, **To Be Handled**, **Handled by Me**, and **My Permissions** pages respectively.

Submitted by Me

In the left-side navigation pane, click **Submitted by Me**.

On the **Submitted by Me** page, you can view request details or cancel requests submitted by you. To resubmit a request that was not approved, find the target request and click **Reapply**.

To Be Handled

In the left-side navigation pane, click **To Be Handled**. On the **To Be Handled** page, you can view request details and approve or reject requests.

Handled by Me

In the left-side navigation pane, click **Handled by Me** to view requests handled by you.

My Permissions

In the left-side navigation pane, click **My Permissions**. On the **My Permissions** page, you can view your permissions on tables, files, and APIs respectively.

2.12.6. Perform cross-tenant authorization

Authorization logic

The logic of authorization within a tenant and that of cross-tenant authorization are as follows:

- **Authorization within a tenant:** An access control list (ACL) is used for authorization. You need to add the applicant to the corresponding workspace and then grant permissions to the applicant as requested.
- **Cross-tenant authorization:** A package is used for authorization. First, check whether the workspace where the requested resources reside has a package.
 - If the workspace does not have a package, create a package and add the requested resources to the package. Then, install the package in the workspace where the requested resources are used.
After that, use an ACL to grant permissions to the applicant as requested.
 - If the workspace has a package, add the requested resources to the package.

If the requested resources need to be shared with multiple workspaces, install the package in all these workspaces. If the workspace where the requested resources reside has multiple packages, install all of them in the workspaces where the requested resources are used.

Procedure

1. Log on to the DataWorks console. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Asset Management**.
2. On the **Asset Portal** tab, enter the name of the table that belongs to another tenant in the search box and click **Search**.
3. On the **Asset Category** tab that appears, click the name of the found table on the **Tables** tab. The details page of the table appears.

On the details page, you can view details about the table, including the basic information, business information, physical information, and field information.

4. Click **Request Permission** in the upper-right corner. The **Table Permission Request** page appears.
5. Set parameters on the **Table Permission Request** page.

Parameter	Description
Target Workspace	The workspace where the table is to be used.
Environment	The type of the environment where the table is to be used. If the selected workspace is in standard mode, you need to set this parameter to Development or Production .
Home Workspace	The workspace where the table resides. This parameter is automatically set.
Home Tenant	The tenant to which the table belongs. This parameter is automatically set.
Grant To	The account for which permissions on the table are requested. Valid values: Current Account and System Account for Production Environment .
Requested Period	The period in which the requested permissions are valid.
Reason for Request	The reason why you request the permissions. To improve the review efficiency, we recommend that you describe the reason in detail.
Objects Requested	The tables you want to use. By default, the current table is selected. You can select other tables.

6. After the configuration is completed, click **Submit**.

After you submit the request, click **Approval Management** in the top navigation bar and then click **Submitted by Me**. You can view the review progress on the page that appears.

- If you want to revoke a request, click **Revoke** in the Actions column of the request.
- If you want to submit a revoked or rejected request again, click **Reapply** in the Actions column of the request.

7. Log on to the DataWorks console as the table owner, go to the **Approval Management** tab and click

To Be Handled. On the page that appears, review the request information and click **Agree**.

After clicking **Agree**, click **Approval Management** in the top navigation bar and click **Handled by Me**. On the page that appears, you can view your authorization records and revoke specific authorizations.

2.13. Organization management

2.13.1. Member management

1. [Log on to the DataWorks console](#) as a workspace administrator.
2. Move your pointer over the DataWorks icon in the upper-left corner, and click **Project Management**.
3. Click **Member Management** in the left-side navigation bar. The **Members** page appears.
4. You can enter a member name or logon name in the search box to search for a member to be added or removed from the current organization.
 - Assign a role
To assign a role to a member, click the **Roles** drop-down list next to the member, and select the role to be assigned.
To unassign a role from a member, click **x** next to the role.
 - Remove a member from an organization
Click **Delete** next to the member, and click **OK** in the **Remove from Tenant** dialog box that appears.

2.13.2. Resource groups

2.13.2.1. Go to the page for managing scheduling resources

Scheduling resources are managed at the tenant level. An exclusive scheduling resource can consist of multiple physical machines for running nodes.

1. [Log on to the DataWorks console](#).
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Project Management**.
3. In the left-side navigation pane, click **Schedule Resources**. On the Schedule Resources page, manage scheduling resources as required.

 **Note** If you are the tenant administrator, you can modify existing scheduling resources on the Schedule Resources page.

2.13.2.2. Change the workspace to which a scheduling resource belongs

After you create and configure an exclusive scheduling resource, you can change the workspace to which the resource belongs.

Procedure

1. [Log on to the DataWorks console](#) as the tenant administrator.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > Project Management**.
3. On the page that appears, click **Schedule Resources** in the left-side navigation pane. On the **Schedule Resources** page, enter a keyword in the search box in the upper-left corner. DataWorks searches for the

scheduling resources whose names contain the keyword in fuzzy match mode.

4. Find the target scheduling resource and click **Modify attribution**.
5. In the **Modify attribution** dialog box, select the workspace to which the exclusive scheduling resource will be bound.
6. Click **OK**.

2.13.3. Configure the compute engine

Currently, DataWorks only supports MaxCompute as its compute engine. All business flows and nodes in a workspace are run on the MaxCompute project associated to the workspace.

Example

 **Note** Tenant administrators can modify the settings for MaxCompute projects. The following settings are changeable: the project description, whether to use the MaxCompute project owner account to run MaxCompute jobs, the account used for running MaxCompute jobs, and the AccessKey of the account.

Assume that the account used for running MaxCompute jobs is no longer available, for example, because the account owner has resigned. If **Run MaxCompute Task Using MaxCompute Owner Account** is not selected, the tenant administrator needs to immediately modify the account used for running MaxCompute jobs and its AccessKey so that tasks can properly run in the workspace that uses the corresponding MaxCompute project.

Procedure

You can modify the account used for running MaxCompute jobs and its AccessKey as follows:

1. Log on to the DataWorks console as a tenant administrator. For more information, see [Log on to the DataWorks console](#).
2. Choose **Project Management > Compute Engine**.
3. In the search box on the **Project Management > Compute Engine** page, enter the compute engine name. Fuzzy search is supported.
4. Find the target compute engine, and click **Configure** in the Actions column.
5. In the **Configure Compute Engine** dialog box, specify the Alibaba Cloud Account and the AccessKey.

 **Note** You can also select **Run MaxCompute Task Using MaxCompute Owner Account** or create a new Alibaba Cloud account.

6. Click **Submit**.

2.14. Data Service

2.14.1. Overview

With Data Service, you can manage all your table APIs after you create new APIs or register existing APIs. You can also easily publish your APIs to API Gateway. Together with API Gateway, Data Service provides a secure, stable, low-cost, and easy-to-use data sharing service.

Data Service adopts a serverless architecture and allows you to develop table APIs without thinking about infrastructure such as compute resources. Data Service supports automatic scaling for compute resources, which significantly reduces your OPEX.

Create an API

In Data Service, you can quickly create APIs based on tables in relational databases or NoSQL databases using a visual wizard. It takes only a few minutes to configure a data API, and coding is not required. You can also create APIs by specifying SQL scripts. The script mode supports advanced functions such as associative tables, complex criteria, and aggregate functions.

Register an API

You can register existing RESTful APIs to Data Service for unified API management. Four request methods and three data formats are supported. The four request methods are GET, POST, PUT, and DELETE. The three data formats are tables, JSON, and XML.

API Gateway

API Gateway provides API lifecycle management services, including API publishing, management, maintenance, and monetization. It enables low-risk, simple, cost-effective, and fast microservice integration, front and back end separation, and system integration. You can use API Gateway to share functions and data with your partners and third-party developers.

API Gateway supports authorization, authentication, flow control, and billing for Data Service.

2.14.2. Terms

This section introduces terms of Data Service.

Name	Description
Data source	Indicates database links. Data Service accesses data through data sources. Data sources are configured in Data Integration.
Create an API	Creates APIs based on data tables.
Register an API	Registers existing APIs to Data Service for unified management.
Wizard mode	Guides you through the procedure of API creation. This method is suitable for beginners who want to create simple APIs. You do not need to write any code.
Script mode	Allows you to create APIs by writing SQL scripts. This method supports associative tables, complex queries, and aggregate functions. This method is suitable for experienced developers who want to create complex APIs.
API group	Indicates a set of APIs for a specific scenario or for consuming a specific service. An API group is the smallest group unit in Data Service, and the smallest unit for API Gateway management. API groups are published in Alibaba Cloud API Marketplace as API products.
API Gateway	Indicates a hosted service provided by Alibaba Cloud to manage APIs. API Gateway supports API lifecycle management, permission management, access management, and traffic control.

2.14.3. Manage tags

This topic describes how to create, add, view, and remove tags for an API.

DataService Studio allows you to add tags to APIs when you manage workflows and create, register, and deploy APIs. This topic uses the scenario of creating an API in the codeless UI as an example. Tags allow you to efficiently classify and search for APIs. You can maintain only one tag list in a workspace, and cannot use the tag list of a workspace in another workspace.

 **Note**

- Each API supports zero to five tags. That is, you can add no tags to an API at all or add a maximum of five tags to an API.
- The name of a tag can contain letters, digits, and underscores (_), and cannot exceed 20 characters in length.

Create a tag for an API

1. Log on to the DataWorks console, click the DataWorks icon in the upper-left corner, and then choose **All Products > Data Service**.
2. On the Service Development tab, move the pointer over  and choose **API > Generate API**.
3. In the **Generate API** dialog box, set the API mode parameter to Wizard Mode and enter a tag in the **Label** field.
If the tag you entered does not exist in the current workspace, Add <Tag> appears in the drop-down list. Click Add <Tag> to create the tag for the API.
4. Set other parameters and click **OK**. The tag is created for the API and appears in the tag list of the current workspace.

Add an existing tag to an API

1. Log on to the DataWorks console, click the DataWorks icon in the upper-left corner, and then choose **All Products > Data Service**.
2. On the Service Development tab, move the pointer over  and choose **API > Generate API**.
3. In the **Generate API** dialog box, set the API mode parameter to Wizard Mode and click a blank area or the downward arrow in the **Label** field. Tags in the current workspace appear in a drop-down list.
4. Click the required tag, set other parameters, and then click **OK**. The tag is added to the API.

View tags of an API

1. Log on to the DataWorks console, click the DataWorks icon in the upper-left corner, and then choose **All Products > Data Service**.
2. On the **Service Development** tab, double-click the name of the target API in the API list.
3. In the right-side navigation pane, click **Properties**. Then you can view the tags of the API in the **Label** column.

 **Note** You can also create, add, and remove tags for the API in the Properties pane.

If an API is published, you can also perform the following steps to view the tags of the API:

- i. In DataService Studio, click **Service Management** in the upper-right corner.
- ii. Click **Manage APIs** in the left-side navigation pane. On the page that appears, click the **APIs of Published** tab and click the name of the target API. The **API Details** page appears.
- iii. View the tags of the API under **Label** in the **API Basic Information** section.

Remove a tag from an API

1. Log on to the DataWorks console, click the DataWorks icon in the upper-left corner, and then choose **All Products > Data Service**.

2. On the **Service Development** tab, double-click the name of the target API in the API list.
3. In the right-side navigation pane, click **Properties**. Then, you can view the tags of the API in the **Label** column.
4. Click  next to the tag to remove.
5. Click **Publish** in the upper-right corner. The tag is removed from the API.

Search for APIs by tag

1. Log on to the DataWorks console, click the DataWorks icon in the upper-left corner, and then choose **All Products > Data Service**.
2. On the **Service Development** tab, click **Service Management** in the upper-right corner.
3. Click **Manage APIs** in the left-side navigation pane. On the page that appears, click the **APIs of Published** tab and view the tags of each API in the **Label** column.
If an API has multiple tags and some of them are hidden, click ... to show all the tags.
4. On the **APIs of Published** tab, click **Advanced Search**.
5. Enter a tag in the **Label** field to search for all APIs associated with the tag.

 **Note** You can search for APIs based on multiple tags.

2.14.4. Manage business processes and objects under business processes

2.14.4.1. Manage business processes

This topic describes how to create, modify, and delete a business process.

Context

DataWorks allows you to organize different types of resources in a business process. This helps you analyze data by business. Each business process contains APIs, functions, and workflows.

Create a business process

1. [Log on to the DataWorks console](#).
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, move the pointer over  and select **Business process**.
4. In the **New business process** dialog box, set the parameters as required.

New business process ✕

i An API Group is an API Gateway unit that manages APIs. All APIs under a business process belong to the API Group specified by the business process.

* Business Name : 0/50

The business name must be unique. It can contain 4 to 50 characters, including Chinese characters, English letters, numbers, and underscores in English format. It must start with an English letter or Chinese character.

* API grouping : Please Select ▾

To create a new group, you can jump [API Gateway](#) Create new grouping

Business Description : 0/180

Business Description, no more than 180 characters

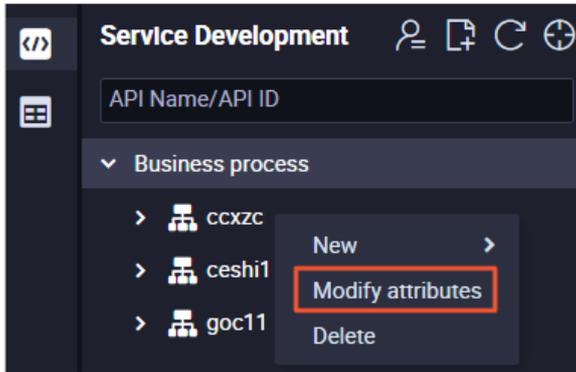
OK
Cancel

Parameter	Description
Business Name	<p>The name of the business process.</p> <ul style="list-style-type: none"> ◦ The name can contain letters, digits, and underscores (_). ◦ The name must start with a letter. ◦ The name must be 4 to 50 characters in length. ◦ The name must be unique in the workspace to which the business process belongs.
API Group	<p>The API group to which the APIs under the business process belong. An API group is the API management unit of API Gateway.</p> <p>You can select an API group from the drop-down list.</p> <p>DataService Studio creates a default API group named in the Workspace name_default format for each user. If you need to create an API group, click API Gateway in the note to go to the API Gateway console and create an API group.</p>
Business Description	<p>The description of the business process. The description can be up to 180 characters in length.</p>

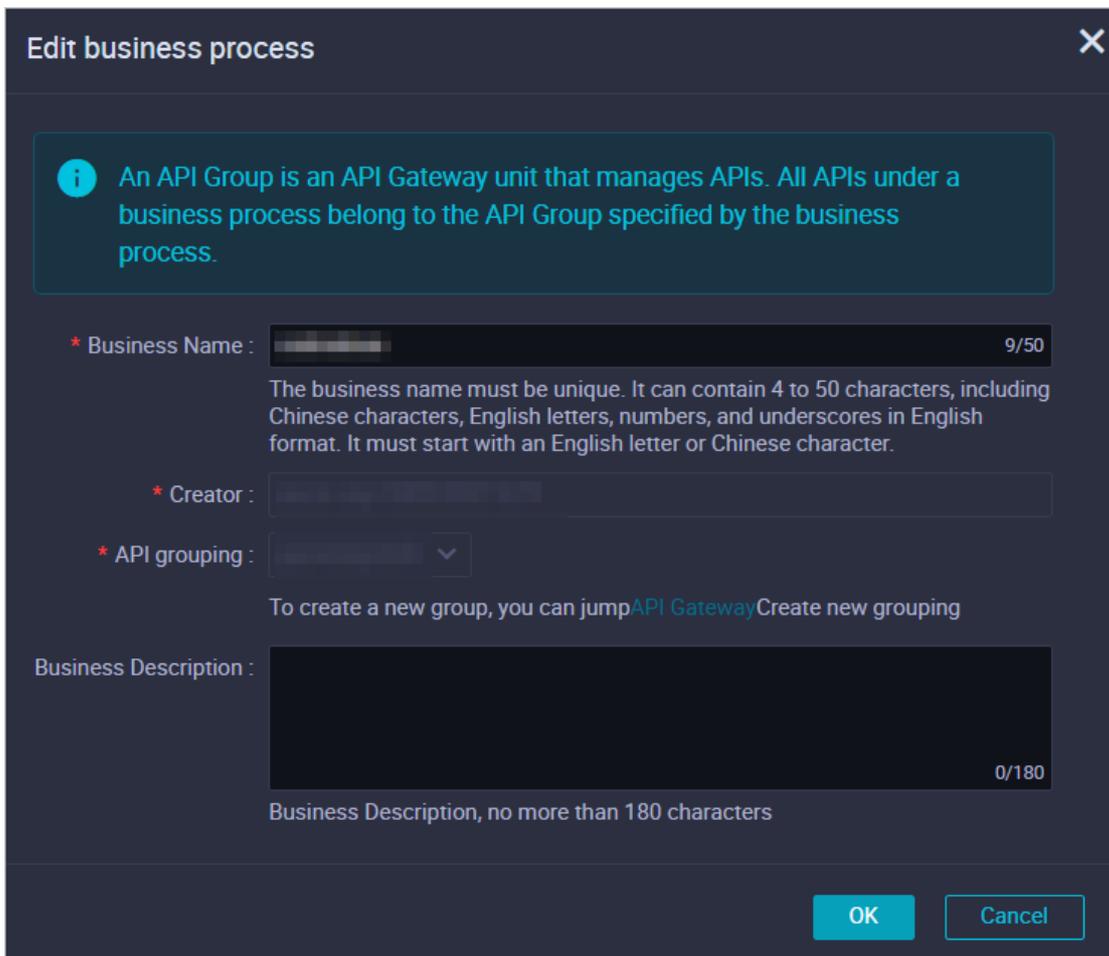
5. Click **OK**. After the business process is created, you can view it in the business process list.

Modify a business process

1. On the **Service Development** tab, right-click the name of the target business process and select **Modify attributes**.



2. In the **Edit business process** dialog box, modify the **Business Name** and **Business Description** parameters as required.

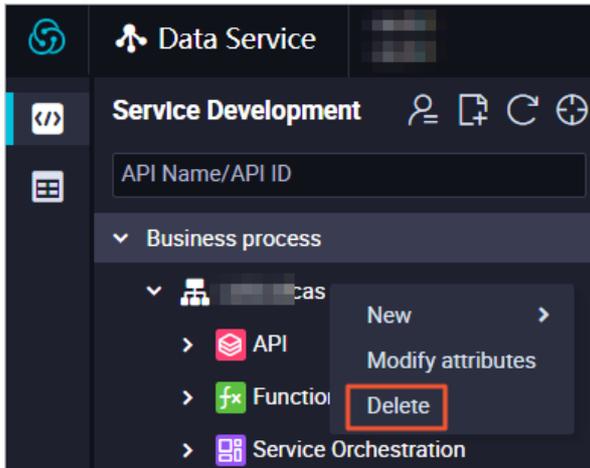


Note You cannot modify the **Creator** or **API Group** parameter of a business process.

3. Click **OK**.

Delete a business process

1. On the **Service Development** tab, right-click the name of the target business process and select **Delete**.



2. In the message that appears, click OK.

Note

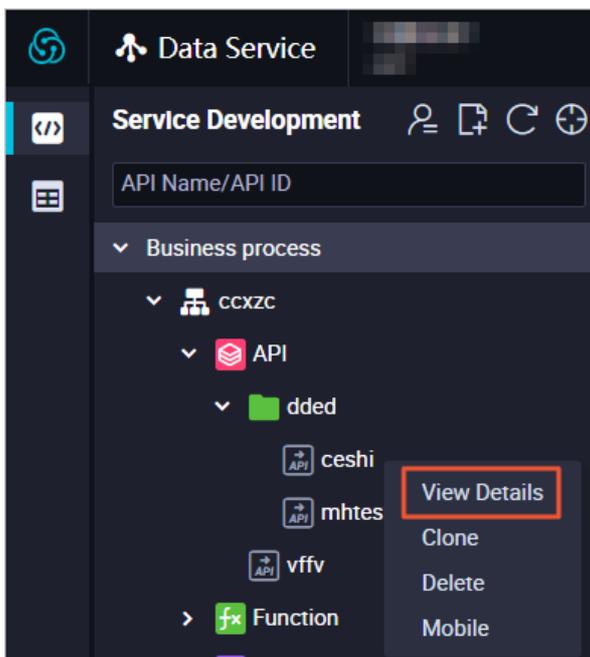
- You can delete only business processes that do not contain any objects such as APIs, functions, or workflows.
- If you need to delete a business process that contains such objects, delete the objects before you delete the business process.

2.14.4.2. Manage APIs

This topic describes how to view, clone, delete, and move APIs.

View an API

1. [Log on to the DataWorks console.](#)
2. Click  in the upper-left corner and choose **All Products > Data Service.**
3. On the **Service Development** tab, right-click the name of the target API and select **View Details.**



Note The **View Details** option appears only in the shortcut menu of an API that has been published. If an API has not been published, double-click the API to go to the configuration tab of the API. Then, click **Properties** in the right-side navigation pane to view its basic information.

Clone an API

You can clone an API to a specified directory in the directory tree.

1. On the **Service Development** tab, right-click the name of the target API and select **Clone**.
2. In the **Clone API** dialog box, set the parameters as required.

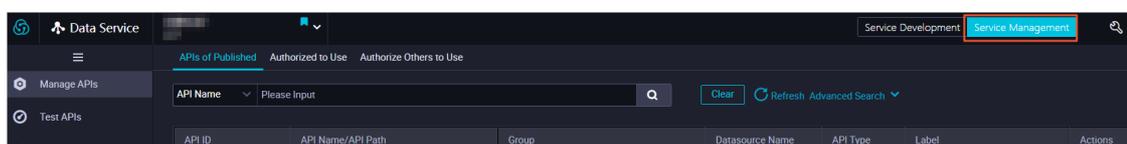
Parameter	Description
API Name	The name of the cloned API. It must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.
API Path	The path for storing the cloned API, for example, <i>/user</i> . The path can contain letters, digits, underscores (_), and hyphens (-). It must start with a forward slash (/) and can be up to 200 characters in length.
Target Folder	The directory for storing the cloned API.

3. Click **OK**.

Delete an API

You can delete only APIs that have not been published. To delete APIs that have been published, you must unpublish them first.

1. Optional. Unpublish the target API. If the API to be deleted is in the Unpublished state, skip this step.
 - i. Go to the **Service Development** tab and click **Service Management** in the upper-right corner.



- ii. On the page that appears, click the **APIs of Published** tab, find the target API, and then click **Unpublish** in the Actions column.
 - iii. In the **Unpublish API** message, click **OK**.
 - iv. Click **Service Development** in the upper-right corner to return to the **Service Development** tab.
2. On the **Service Development** tab, right-click the name of the target API and select **Delete**.
 3. In the **Delete API** message, click **OK**.

Note Deleted APIs cannot be recovered. Use caution when you delete an API.

Move an API to another directory

You can move only APIs that have not been published. To move APIs that have been published, you must unpublish them first.

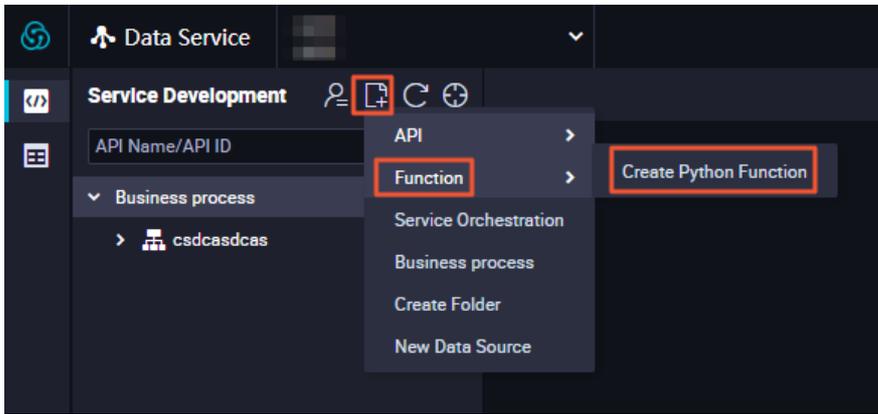
1. On the **Service Development** tab, right-click the name of the target API and select **Mobile**.
2. In the **Modify file path** dialog box, set the **Target Folder** parameter.
3. Click **OK**.

2.14.4.3. Manage functions

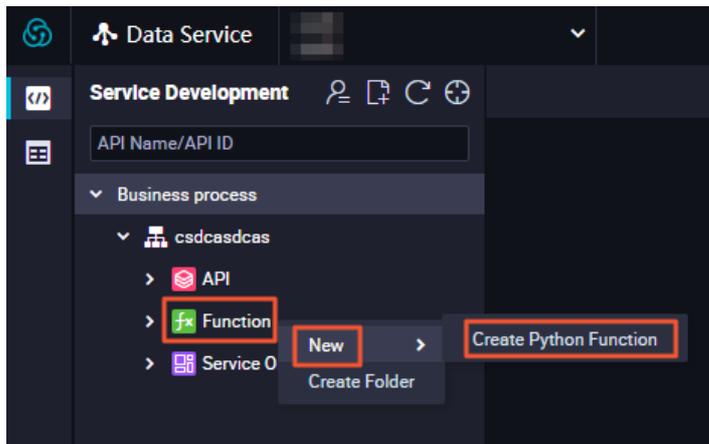
This topic describes how to create, clone, delete, and move Python functions.

Create a function

1. [Log on to the DataWorks console](#).
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. Move the pointer over  and choose **Function > Create Python Function**.

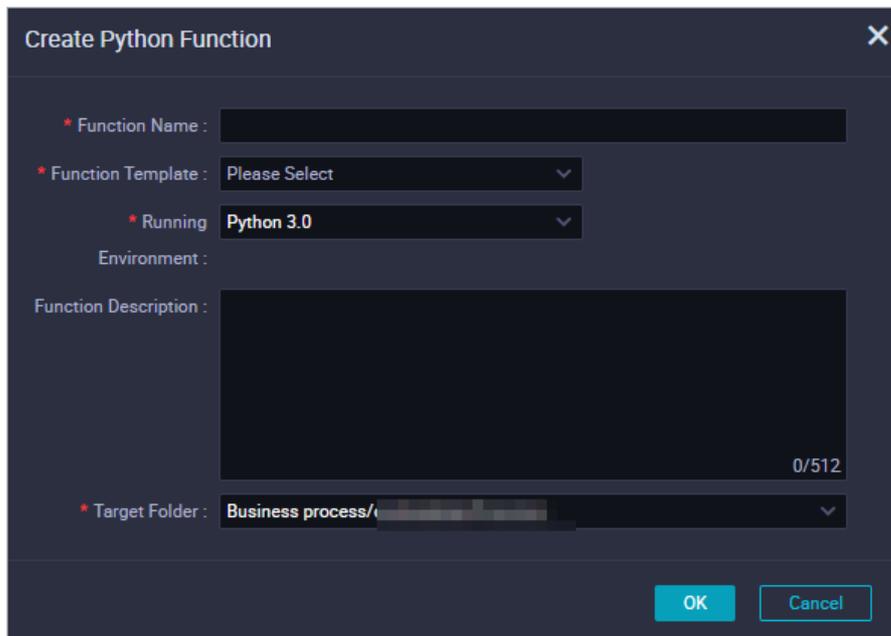


You can also click a business process, right-click **Function**, and then choose **New > Create Python Function**.



Notice DataService Studio allows you to create only Python functions.

4. In the **Create Python Function** dialog box, set the parameters as required.



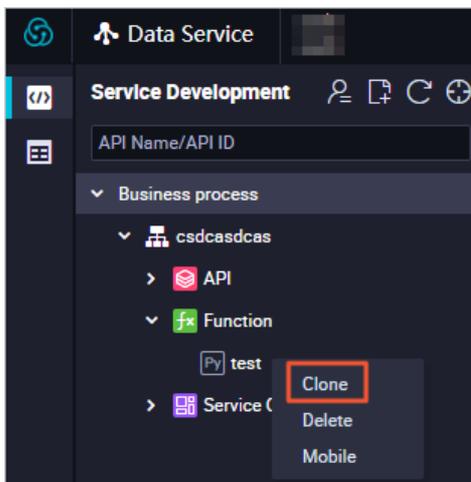
Parameter	Description
Function Name	The name of the function to create, which can be up to 256 characters in length.
Function Template	The template used to create the function. Set the value to Python3 Standard v1.
Running Environment	The runtime environment of the function. Set the value to Python 3.0.
Function Description	The description of the function. The description can be up to 512 characters in length.

Parameter	Description
Target Folder	The directory for storing the function.

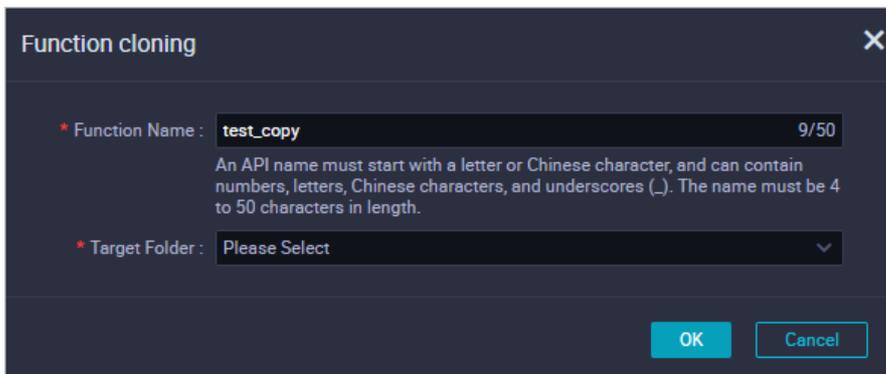
5. Click OK.
6. On the configuration tab of the function, configure the function.
 - i. In the **Edit Code** section, enter the function code.
 - ii. In the **Environment Configuration** section, set the **Memory** and **Function Timeout** parameters.
7. Click Save icon in the toolbar.

Clone a function

1. On the **Service Development** tab, right-click the name of the target function and select **Clone**.



2. In the **Function cloning** dialog box, set the **Function Name** and **Target Folder** parameters.



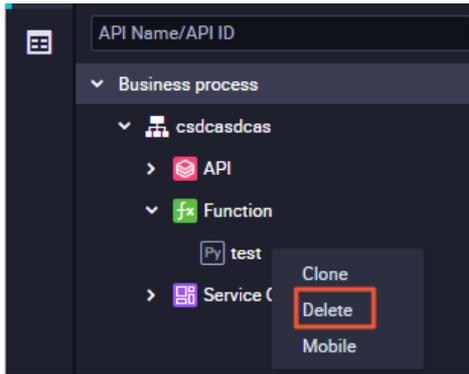
Note The name of the function must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.

3. Click OK.

Delete a function

1. On the **Service Development** tab, right-click the name of the target function and select **Delete**.



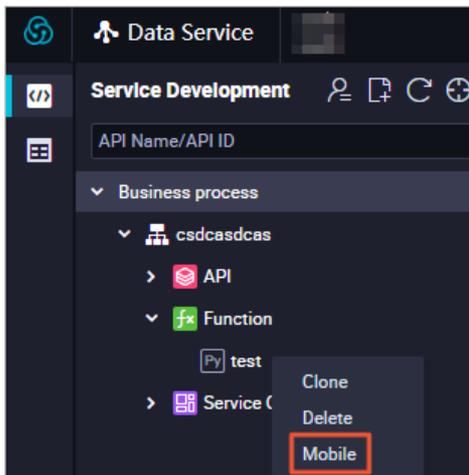


2. In the message that appears, click OK.

Note You can delete only functions that are not referenced by APIs. You must remove the function from the filters of the APIs that reference the function before you can delete the function.

Move a function to another directory

1. On the **Service Development** tab, right-click the name of the target function and select **Mobile**.



2. In the **Modify file path** dialog box, set the **Target Folder** parameter.
3. Click OK.

2.14.4.4. Manage workflows

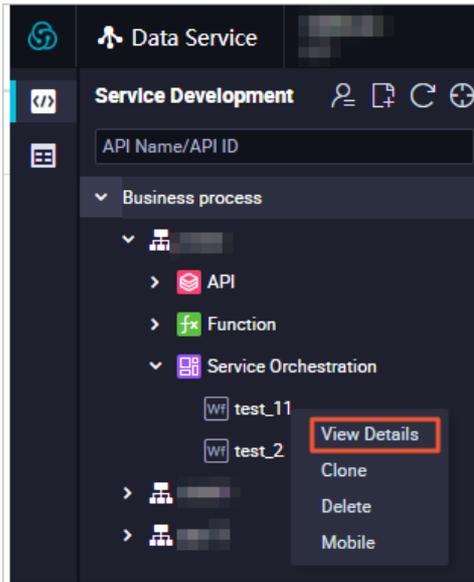
This topic describes how to view, clone, delete, and move a workflow.

Prerequisites

Workflows are created and published. For more information, see [Use workflows](#).

View a workflow

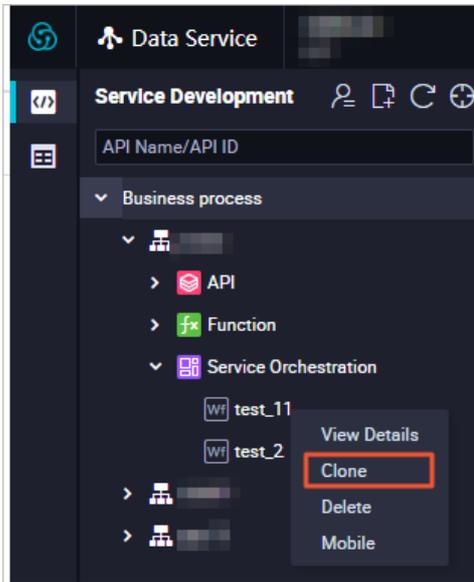
1. [Log on to the DataWorks console](#).
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, right-click the name of the target workflow and select **View Details**.



Note The View Details option appears only in the shortcut menu of a workflow that has been published. If a workflow has not been published, double-click the workflow to go to the configuration tab of the workflow. Then, click **Properties** in the right-side navigation pane to view its basic information.

Clone a workflow

1. On the **Service Development** tab, right-click the name of the target workflow and select **Clone**.



2. In the **Clone API** dialog box, set the parameters as required.

Parameter	Description
API Name	The name of the cloned workflow. It must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.

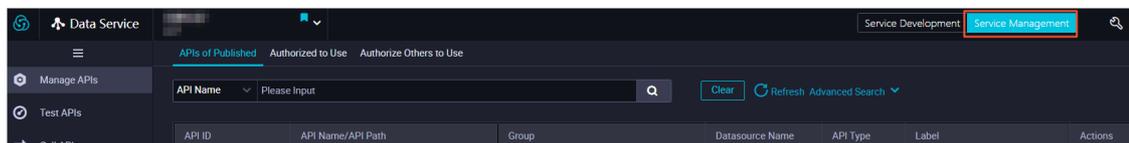
Parameter	Description
API Path	The path for storing the cloned workflow, for example, <i>/user</i> . The path can contain letters, digits, underscores (_), and hyphens (-). It must start with a forward slash (/) and can be up to 200 characters in length.
Target Folder	The directory for storing the cloned workflow.

3. Click **OK**.

Delete a workflow

You can delete only workflows that have not been published. To delete workflows that have been published, you must unpublish them first.

1. Optional. Unpublish the target workflow. If the workflow to be deleted is in the Unpublished state, skip this step.
 - i. Go to the **Service Development** tab and click **Service Management** in the upper-right corner.



- ii. On the page that appears, click the **APIs of Published** tab, find the target API, and then click **Unpublish** in the Actions column.
 - iii. In the **Unpublish API** message, click **OK**.
 - iv. Click **Service Development** in the upper-right corner to return to the **Service Development** tab.
2. On the **Service Development** tab, right-click the name of the target workflow and select **Delete**.
 3. In the **Delete API** message, click **OK**.

Note Deleted workflows cannot be recovered. Use caution when you delete a workflow.

Move a workflow to another directory

You can move only workflows that have not been published. To move workflows that have been published, you must unpublish them first.

1. On the **Service Development** tab, right-click the name of the target workflow and select **Mobile**.
2. In the **Modify file path** dialog box, set the **Target Folder** parameter.
3. Click **OK**.

2.14.5. Create an API

In Data Service, you can quickly create APIs based on tables in relational databases or NoSQL databases using a visual wizard. It takes only a few minutes to configure a data API, and coding is not required.

You can also create APIs by specifying SQL scripts. The script mode supports advanced functions such as associative tables, complex criteria, and aggregate functions.

The differences between the wizard mode and script mode are described as follows:

Differences between the wizard mode and script mode

Category	Description	Wizard mode	Script mode
Query object	Queries a single table from one data source	Supported	Supported
	Queries associative tables from one data source	Not supported	Supported
Search condition	Searches for an exact number	Supported	Supported
	Searches for a range of numbers	Not supported	Supported
	Matches an exact string	Supported	Supported
	Performs fuzzy search for strings	Supported	Supported
	Sets required and optional parameters	Supported	Supported
Query result	Returns the field value	Supported	Supported
	Performs a mathematical calculation for field values	Not supported	Supported
	Performs an aggregate operation on field values	Not supported	Supported
	Displays results with pagination	Supported	Supported

2.14.5.1. Configure connections

DataService Studio allows you to obtain table schemas and query data through APIs from connections.

 **Note** Before generating an API, make sure that you have configured the relevant connections.

You can click **Connections** on the **Data Integration** page to configure connections. The following table lists the available connection types and supported configuration modes.

Connection name	Create an API in the codeless UI	Create an API in the code editor
ApsaraDB for RDS	Supported	Supported
DRDS	Supported	Supported
ApsaraDB for MySQL	Supported	Supported
ApsaraDB for PostgreSQL	Supported	Supported
ApsaraDB for SQL Server	Supported	Supported
Oracle	Supported	Supported

Connection name	Create an API in the codeless UI	Create an API in the code editor
AnalyticDB	Supported	Supported
Table Store	Supported	Not supported
MongoDB	Supported	Not supported

2.14.5.2. Create an API in the codeless UI

DataWorks allows you to create APIs by setting parameters in the codeless UI without the need to write code. This topic describes how to create an API in the codeless UI.

Prerequisites

Connections are configured on the **Data Source** page. For more information, see [Configure data sources](#).

Create an API

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, move the pointer over  and choose **API > Generate API**. You can also click a business process, right-click **API**, and then choose **New > Generate API**.
4. In the **Generate API** dialog box, set the parameters as required.

Parameter	Description
API mode	The mode for creating the API. Valid values: Wizard Mode and Script Mode . In this example, select Wizard Mode .
API Name	The name of the API. The name must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.
API Path	The path for storing the API, for example, <i>/user</i> .
Call Mode	The mode for calling the API. Valid values: Synchronous Call and Asynchronous Call . <ul style="list-style-type: none"> ◦ If you set this parameter to Synchronous Mode, the API returns results immediately after it is called. The synchronous mode is most commonly used. ◦ If you set this parameter to Asynchronous Mode, the API returns the RequestID parameter immediately after it is called. The API caller can then obtain the call result from a message queue based on the request ID.
Protocol	The protocol used by the API. Valid values: HTTP and HTTPS .
Request Method	The request method used by the API. Valid values: GET and POST .

Parameter	Description
Response Content Type	The return type of the API. Set the value to JSON .
Visible Range	<p>The visibility of the API. Valid values:</p> <ul style="list-style-type: none"> ◦ Work Space: The API is visible to all members in the current workspace. ◦ Private: The API is visible only to its owner and permissions on the API cannot be granted to other users. <p>Note If you set the Visible Range parameter to Private, the API is visible only to you in the directory tree. It is hidden to other members of the workspace.</p>
Label	<p>The tag of the API. Select one or more tags from the drop-down list. For more information, see Manage tags.</p> <p>Note You can set at most five tags for an API.</p>
Description	The description of the API, which can be up to 2,000 characters in length.
Target Folder	The directory for storing the API.

5. Click **OK**.

Configure the API

1. Double-click the API in the directory tree. On the configuration tab that appears, set the **DataSource Type**, **DataSource Name**, and **Table Name** parameters in the **Select Table** section.

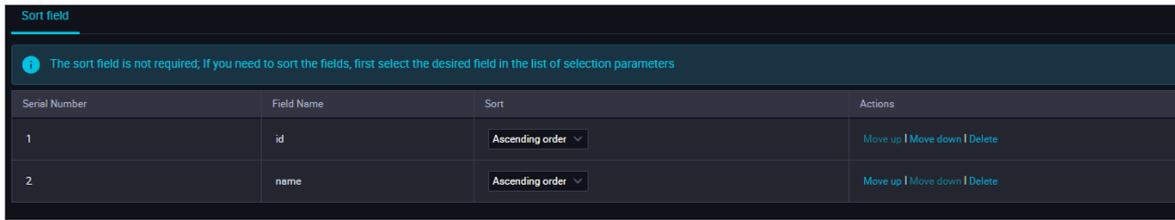
Note

- Before you select a table for an API, you must configure a connection in Data Integration. You can enter a table name in the Table Name field to search for the desired table.
- After you create an API, the table configuration tab automatically appears for you to select a table for the API.

2. In the **Environment Configuration** section, set the **Memory** and **Function Timeout** parameters.
3. In the **Select Parameters** section, set the request and response parameters for the API. After you select a table in the **Select Table** section, all fields in the table appear in the **Select Parameters** section. Select the required fields and select the check boxes in the **Set as Req Param** and **Set as Resp Param** columns as required.

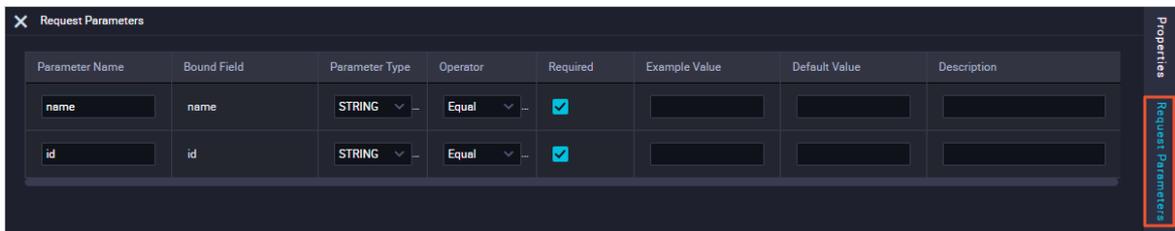
<input type="checkbox"/> Set as Req Param	<input type="checkbox"/> Set as Resp Param	Field Name	Field Type	Field Description	Field sorting
<input checked="" type="checkbox"/>	<input type="checkbox"/>	id	BIGINT UNSIGNED		Add to field sort
<input type="checkbox"/>	<input checked="" type="checkbox"/>	gmt_create	DATETIME		Add to field sort

To sort the data returned by the API based on a field, click **Add to field sort** in the Actions column of the field to add it to the **Sort field** section.



The sorting feature allows you to specify the fields based on which the parameters returned by the API are sorted. A field with a smaller sequence number in the Sort field section has a higher priority in sorting. You can click **Move up** or **Move down** to adjust the sequence of a field. You can specify the sorting mode for each field by selecting **Ascending order** or **Descending order** in the Sort column.

- In the right-side navigation pane, click **Request Parameters**. In the Request Parameters pane, set the parameters as required.



Parameter	Description
Parameter Name	The name of the request parameter. The name can contain letters, digits, underscores (_), and hyphens (-). It must start with a letter and can be up to 64 characters in length.
Bound Field	The field to be bound to the request parameter. You cannot change the value.
Parameter Type	The type of the request parameter. Valid values: STRING , INT , LONG , FLOAT , DOUBLE , and BOOLEAN .

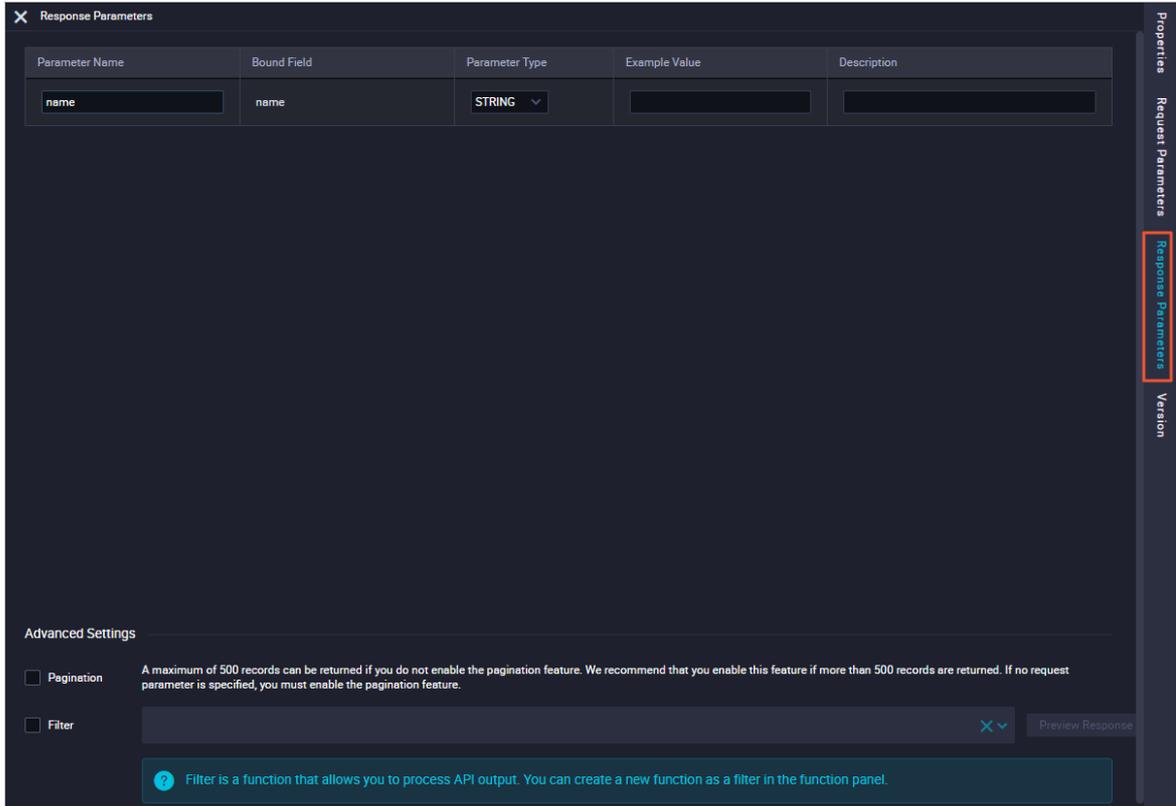
Parameter	Description
Operator	<p>The operator that is used to associate or compare the value of the request parameter with the specified value. You can select one of the following operators:</p> <ul style="list-style-type: none"> ◦ Equal: The value of the request parameter is equal to the specified value. ◦ LIKE: The value of the request parameter matches the specified pattern. ◦ IN: The value of the request parameter is in the specified range. ◦ NOT IN: The value of the request parameter is out of the specified range. ◦ NOT LIKE: The value of the request parameter does not match the specified pattern. ◦ ! =: The value of the request parameter is not equal to the specified value. ◦ >: The value of the request parameter is greater than the specified value. ◦ <: The value of the request parameter is less than the specified value. ◦ >=: The value of the request parameter is greater than or equal to the specified value. ◦ <=: The value of the request parameter is less than or equal to the specified value.
Required	Specifies whether the request parameter is required.
Example Value	The sample value of the request parameter.
Default Value	The default value of the request parameter.
Description	The description of the request parameter.

To preprocess the request parameters of the API, select **Use prefilter** in the **Advanced Settings** section. For more information, see [Use prefilters](#).

 **Note**

- To enhance the matching efficiency, set an indexed field as a request parameter.
- To make it easier for API callers to know the details about the API, we recommend that you specify information such as the sample value, default value, and description for each parameter of the API.

5. In the right-side navigation pane, click **Response Parameters**. In the Response Parameters pane, set the parameters as required.



Parameter	Description
Parameter Name	The name of the response parameter. The name can contain letters, digits, underscores (_), and hyphens (-). It must start with a letter and can be up to 64 characters in length.
Bound Field	The field to be bound to the response parameter. You cannot change the value.
Parameter Type	The type of the response parameter. Valid values: STRING , INT , LONG , FLOAT , DOUBLE , and BOOLEAN .
Example Value	The sample value of the response parameter.
Description	The description of the response parameter.

You can select **Pagination** and **Filter** in the **Advanced Settings** section.

Select **Pagination** based on your needs.

- If you do not select **Pagination**, the API returns a maximum of 2,000 records by default.
- If the API may return more than 2,000 records, we recommend that you select **Pagination**.

The following common parameters are available when **Pagination** is selected:

- Common request parameters
 - **pageNum**: the number of the page to return.
 - **pageSize**: the number of entries to return on each page.
- Common response parameters
 - **pageNum**: the page number of the returned page.

- `pageSize`: the number of entries returned per page.
- `totalNum`: the total number of returned entries.

If you need to process the query results returned by the API, select **Filter**.

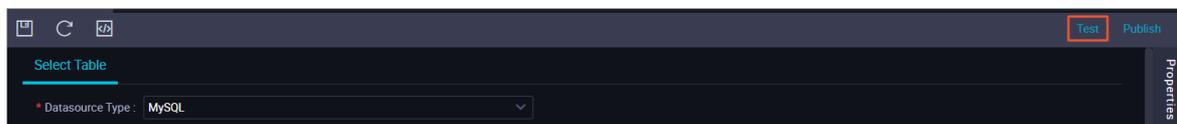
Note

- Field values are returned in the response as they are in the table.
- Request parameters are optional for an API. If you do not specify any request parameters for an API, you must select **Pagination**.

6. Click **Save** icon in the toolbar.

Test the API

1. After you save the settings of the API, click **Test** in the upper-right corner.



2. In the **Test APIs** dialog box, click **Test** to send an API request. The request and response details appear on the right. If the API fails the test, check the error message, modify the API settings accordingly, and test the API again.

You can select **Save the correct response example automatically** as required.

Note

- The system automatically generates sample failure responses and error codes when it tests an API. However, the system does not automatically generate sample success responses. To allow the system to save the success test result as a sample success response, you must select **Save the correct response example automatically** before you perform the test. If the response contains sensitive data that must be de-identified, you can manually edit the response.
- The sample success response is an important reference for API callers, and therefore must be configured.
- The Call Latency value is the latency of the current API request, which is used to evaluate the API performance. If the latency is long, consider optimizing the database.

3. After the API is tested, close the **Test APIs** dialog box and click **Publish** in the upper-right corner of the configuration tab.

Switch from the codeless UI to the code editor

On the configuration tab of an API, you can switch from the codeless UI to the code editor.

1. Go to the **Service Development** tab and double-click the target API. The configuration tab of the API appears.
2. Click  in the toolbar.
3. In the message that appears, click **OK**. Then, you can view the SQL statements of the API in the **Edit query SQL** section.

 Notice

- DataService Studio allows you to switch only from the codeless UI to the code editor.
- After you switch from the codeless UI to the code editor, you cannot switch back to the codeless UI.

2.14.5.3. Create an API in the code editor

To meet the requirements of advanced data query, DataService Studio allows you to create an API by writing an SQL script in the code editor. DataService Studio supports table join queries, complex queries, and aggregate functions. This topic describes how to create an API in the code editor.

Prerequisites

Connections are configured on the **Data Source** page. For more information, see [Configure data sources](#).

Create an API

1. [Log on to the DataWorks console](#).
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, move the pointer over  and choose **API > Generate API**. You can also click a workflow, right-click **API**, and then choose **New > Generate API**.
4. In the **Generate API** dialog box, set the parameters as required.

Parameter	Description
API mode	The mode for creating the API. Valid values: Wizard Mode and Script Mode . In this example, select Script Mode .
API Name	The name of the API. The name must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.
API Path	The path for storing the API, for example, <i>/user</i> .
Call Mode	The mode for calling the API. Valid values: Synchronous Call and Asynchronous Call . <ul style="list-style-type: none">◦ If you set this parameter to Synchronous Mode, the API returns results immediately after it is called. The synchronous mode is most commonly used.◦ If you set this parameter to Asynchronous Mode, the API returns the RequestID parameter immediately after it is called. The API caller can then obtain the call result from a message queue based on the request ID.
Protocol	The protocol used by the API. Valid values: HTTP and HTTPS .
Request Method	The request method used by the API. Valid values: GET and POST .
Response Content Type	The return type of the API. Set the value to JSON .

Parameter	Description
Visible Range	<p>The visibility of the API. Valid values:</p> <ul style="list-style-type: none"> ◦ Work Space: The API is visible to all members in the current workspace. ◦ Private: The API is visible only to its owner and permissions on the API cannot be granted to other users. <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px; margin-top: 10px;"> <p>? Note If you set the Visible Range parameter to Private, the API is visible only to you in the directory tree. It is hidden to other members of the workspace.</p> </div>
Label	<p>The tag of the API. Select one or more tags from the drop-down list. For more information, see Manage tags.</p> <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px; margin-top: 10px;"> <p>? Note You can set at most five tags for an API.</p> </div>
Description	The description of the API, which can be up to 2,000 characters in length.
Target Folder	The directory for storing the API.

5. Click OK.

Configure the API

1. Double-click the API. On the configuration tab that appears, set the **Datasource Type** and **Datasource Name** parameters in the **Select Table** section.

Select Table

* Datasource Type : Select data source type ▼

* Datasource Name : Select data source ▼ ?

? **Note** You must configure a connection in Data Integration in advance. You can enter a keyword in the Table Name field to search for the desired table.

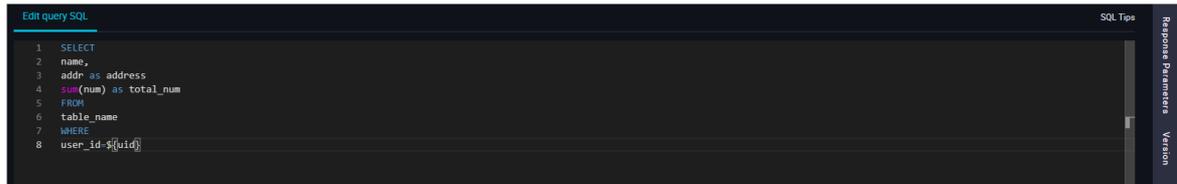
2. In the **Environment Configuration** section, set the **Memory** and **Function Timeout** parameters.

Environment Configuration

* Memory : 4096M ▼

Function Timeout : 10000 ms

3. In the **Edit query SQL** section, enter the SQL statement for querying data.

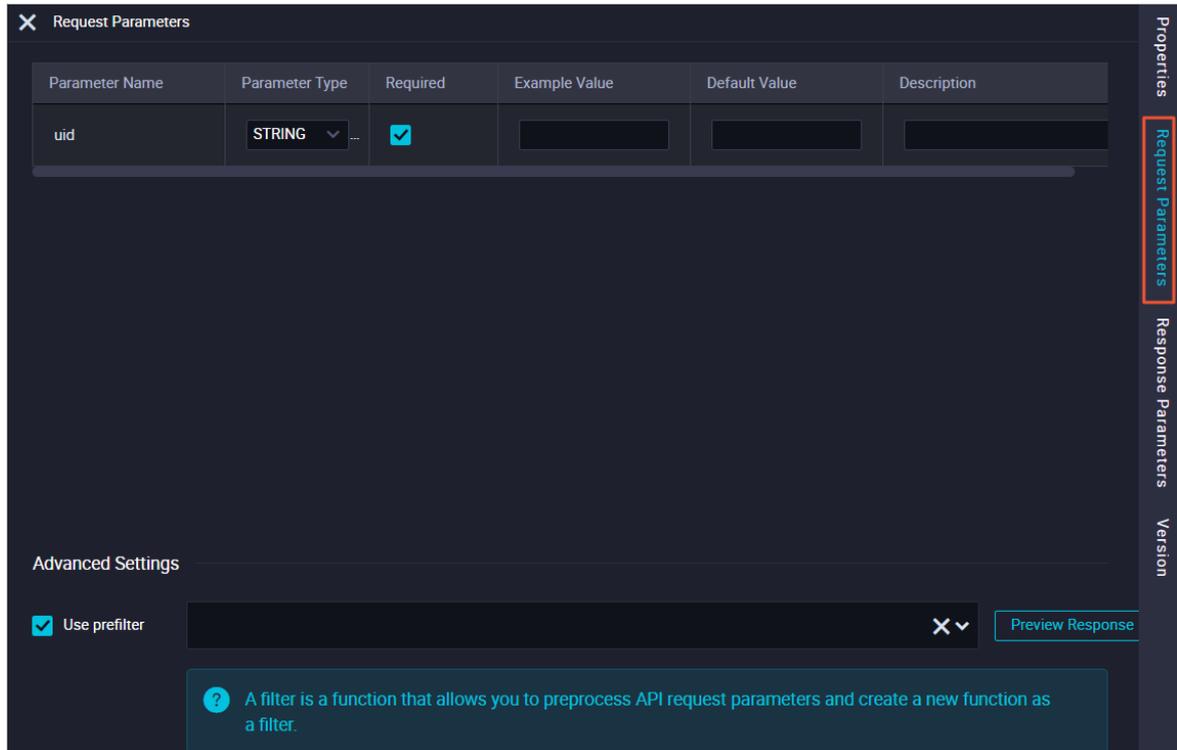


```
1 SELECT
2 name,
3 addr as address
4 sum(num) as total_num
5 FROM
6 table_name
7 WHERE
8 user_id=${uid}
```

Note The SELECT clause specifies the parameters that the API returns. The WHERE clause specifies the request parameters of the API. You must use `${}` to interpolate a request parameter.

Follow these rules when you enter the SQL statement :

- You can enter only one SQL statement in the script editor.
 - Only the SELECT clause is supported. Other clauses such as INSERT , UPDATE, and DELETE are not supported.
 - The clause `SELECT *` is not supported. You must specify the columns to be queried.
 - Single-table queries, table join queries, and nested queries under the same connection are supported.
 - If the name of the column that the SELECT clause specifies has a table prefix, for example, `t.name`, you must create an alias for the corresponding response parameter, for example, `t.name as name`.
 - If you use an aggregate function, such as `min`, `max`, `sum`, or `count`, you must create an alias for the corresponding response parameter, for example, `sum(num) as total_num`.
 - `${param}` in the SQL statement and that in character strings are regarded as request parameters and replaced. If an escape character `\` is placed before `${param}`, `${param}` is processed as a common string.
 - `${param}` cannot be enclosed in single quotation marks (`' '`). For example, `'${id}'` and `'abc${xyz}123'` are not allowed. If necessary, you can use `concat('abc', ${xyz}, '123')` instead.
 - Parameters cannot be configured as optional.
 - `${param}` is not allowed in comments. For example, `--${id}` is not allowed.
4. In the right-side navigation pane, click **Request Parameters**. In the Request Parameters pane, set the parameters as required.



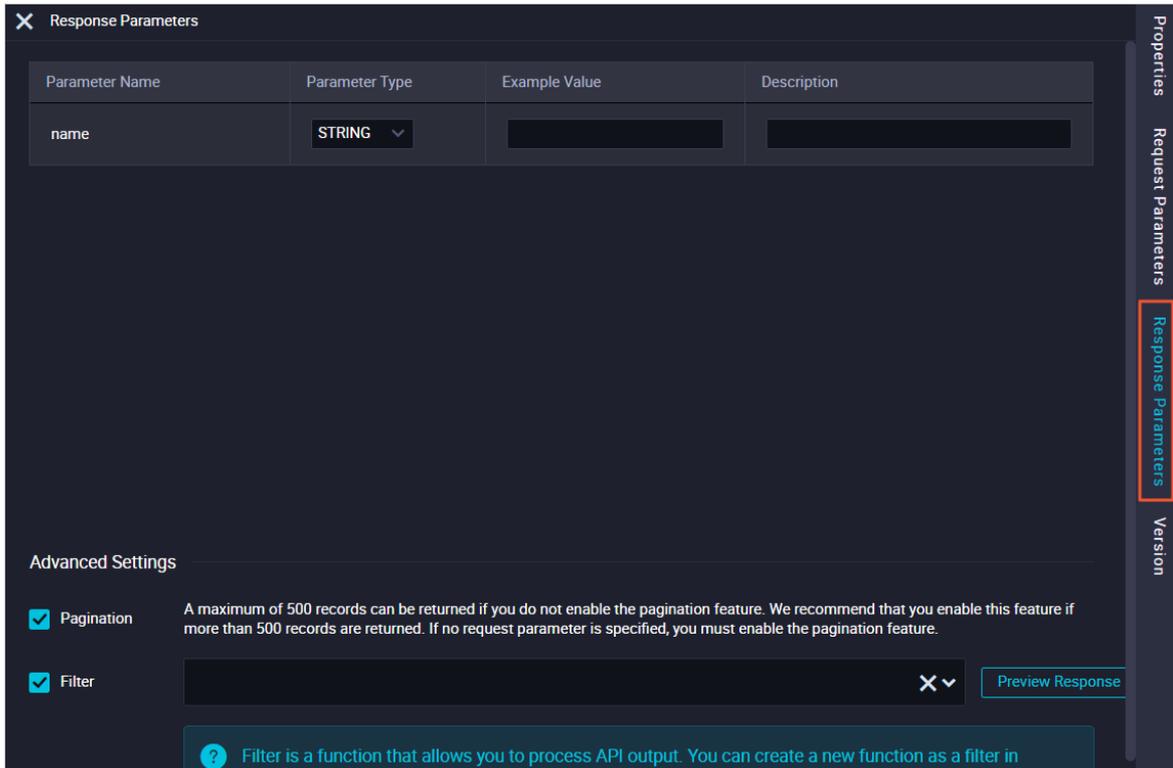
Parameter	Description
Parameter Name	The name of the request parameter. The name can contain letters, digits, underscores (_), and hyphens (-). It must start with a letter and can be up to 64 characters in length.
Parameter Type	The type of the request parameter. Valid values: STRING , INT , LONG , FLOAT , DOUBLE , and BOOLEAN .
Required	Specifies whether the request parameter is required.
Example Value	The sample value of the request parameter.
Default Value	The default value of the request parameter.
Description	The description of the request parameter.

To preprocess the request parameters of the API, select **Use prefilter** in the **Advanced Settings** section. For more information, see [Use prefilters](#).

Note

- To enhance the matching efficiency, set an indexed field as a request parameter.
- To make it easier for API callers to know the details about the API, we recommend that you specify information such as the sample value, default value, and description for each parameter of the API.

- In the right-side navigation pane, click **Response Parameters**. In the Response Parameters pane, set the parameters as required.



Parameter	Description
Parameter Name	The name of the response parameter. The name can contain letters, digits, underscores (_), and hyphens (-). It must start with a letter and can be up to 64 characters in length.
Parameter Type	The type of the response parameter. Valid values: STRING , INT , LONG , FLOAT , DOUBLE , and BOOLEAN .
Example Value	The sample value of the response parameter.
Description	The description of the response parameter.

You can select **Pagination** and **Filter** in the **Advanced Settings** section.

Select **Pagination** based on your needs.

- If you do not select **Pagination**, the API returns a maximum of 2,000 records by default.
- If the API may return more than 2,000 records, we recommend that you select **Pagination**.

The following common parameters are available when **Pagination** is selected:

- Common request parameters
 - **pageNum**: the number of the page to return.
 - **pageSize**: the number of entries to return on each page.
- Common response parameters
 - **pageNum**: the page number of the returned page.
 - **pageSize**: the number of entries returned per page.
 - **totalNum**: the total number of returned entries.

If you need to process the query results returned by the API, select **Filter**.

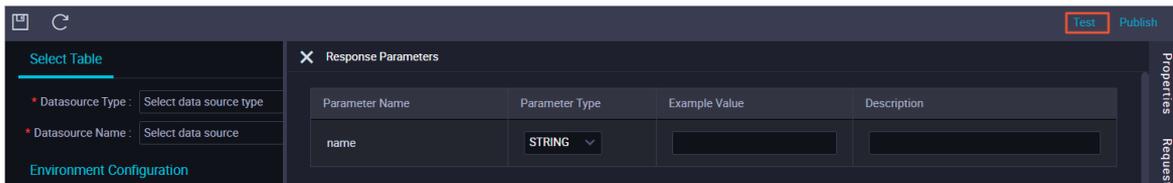
Note

- Field values are returned in the response as they are in the table.
- Request parameters are optional for an API. If you do not specify any request parameters for an API, you must select **Pagination**.

6. Click Save icon in the toolbar.

Test the API

1. After you save the settings of the API, click **Test** in the upper-right corner.



2. In the **Test APIs** dialog box, click **Test** to send an API request. The request and response details appear on the right. If the API fails the test, check the error message, modify the API settings accordingly, and test the API again.

You can select **Save the correct response example automatically** as required.

Note

- The system automatically generates sample failure responses and error codes when it tests an API. However, the system does not automatically generate sample success responses. To allow the system to save the success test result as a sample success response, you must select **Save the correct response example automatically** before you perform the test. If the response contains sensitive data that must be de-identified, you can manually edit the response.
- The sample success response is an important reference for API callers, and therefore must be configured.
- The Call Latency value is the latency of the current API request, which is used to evaluate the API performance. If the latency is long, consider optimizing the database.

3. After the API is tested, close the **Test APIs** dialog box and click **Publish** in the upper-right corner of the configuration tab.

2.14.5.4. Use filters

2.14.5.4.1. Use prefilters

A prefilter is a function that is used to process request parameters of APIs. You can specify one or more prefilters to customize the request content for APIs. This topic describes the limits of prefilters, the built-in function template provided by the system, and how to create functions and use them as prefilters.

Context

Prefilters have the following limits:

- Only Python 3.0 functions can be used as prefilters.
- Prefilters support importing only the following modules: json, time, random, pickle, re, and math.

- The function name of a prefilter must be `def handler(event,context):` .

Function template

The system provides the following built-in function template:

```
# -*- coding: utf-8 -*-
# event (str) : in filter it is the API result, in other cases, it is your param
# context : some environment information, temporarily useless
# import module limit: json,time,random,pickle,re,math
# do not modify function name
import json
def handler(event,context):
# load str to json object
obj = json.loads(event) # Convert the string specified by the event parameter to a JSON object.
# add your code here
# end add
return obj
```

You can modify the function template to write your own function. You can modify the names of the input parameters as needed.

```
Parameter 1 [context]: the context of calling APIs. The value is of the STRING type. This parameter is not in use and is left empty.
Parameter 2 [event]: the result data returned by APIs or the preceding filter. The value is of the STRING type.
```

Create a Python function

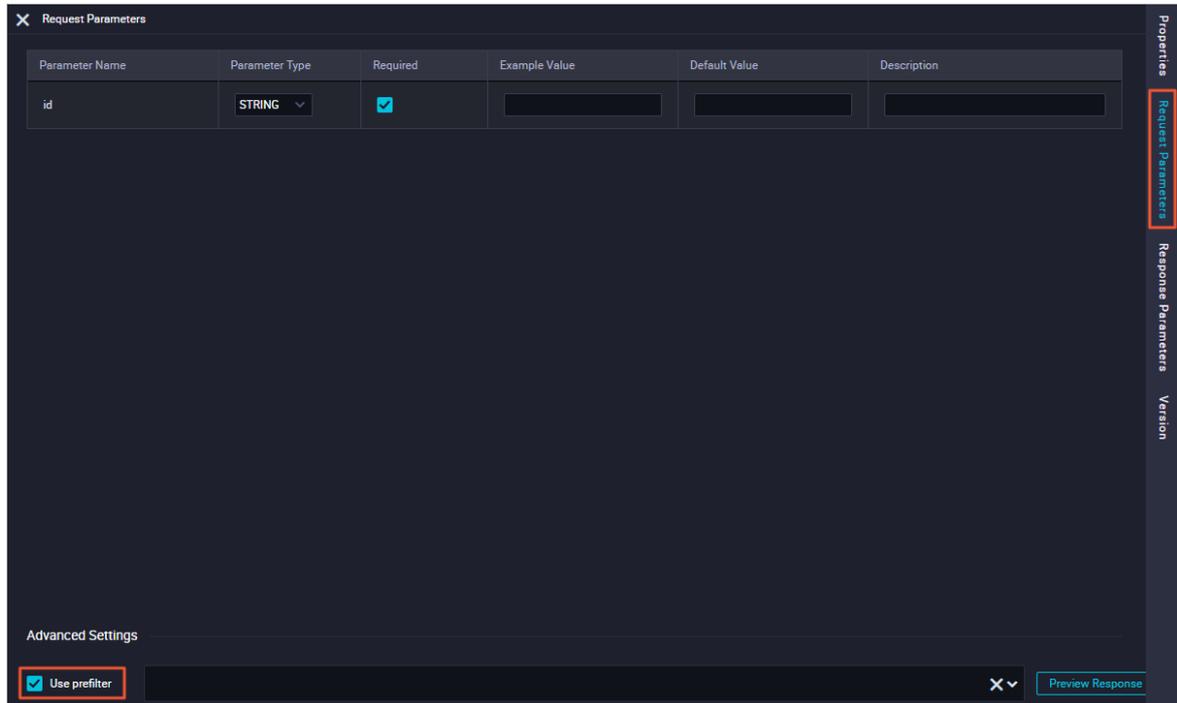
1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, move the pointer over  and choose **Function > Create Python Function**. You can also click a workflow, right-click **Function**, and then choose **New > Create Python Function**.
4. In the **Create Python Function** dialog box, set the parameters as required.

Parameter	Description
Function Name	The name of the function to create, which can be up to 256 characters in length.
Function Template	The template used to create the function. Set the value to Python3 Standard v1.
Running Environment	The runtime environment of the function. Set the value to Python 3.0.
Function Description	The description of the function.
Target Folder	The directory for storing the function.

5. Click **OK**.

Use prefilters

1. On the **Service Development** tab, double-click the target API.
2. On the configuration tab that appears, click **Request Parameters** in the right-side navigation pane.
3. In the **Request Parameters** pane, select **Use prefilter** in the **Advanced Settings** section.



4. Select functions from the **Use prefilter** drop-down list.

Note A prefilter is a function that is used to process request parameters of APIs. You can create a function and use it as a prefilter.

5. Click **Preview Response** to view the processing results of the prefilters.

2.14.5.4.2. Use post filters

A post filter is a function that is used to process the results returned by APIs. You can specify one or more post filters to process the results returned by APIs. This topic describes the limits of post filters, the built-in function template provided by the system, and how to create functions and use them as post filters.

Context

Post filters have the following limits:

- Only Python 3.0 functions can be used as post filters.
- Post filters support importing only the following modules: json, time, random, pickle, re, and math.
- The function name of a post filter must be `def handler(event,context):`.

Function template

The system provides the following built-in function template:

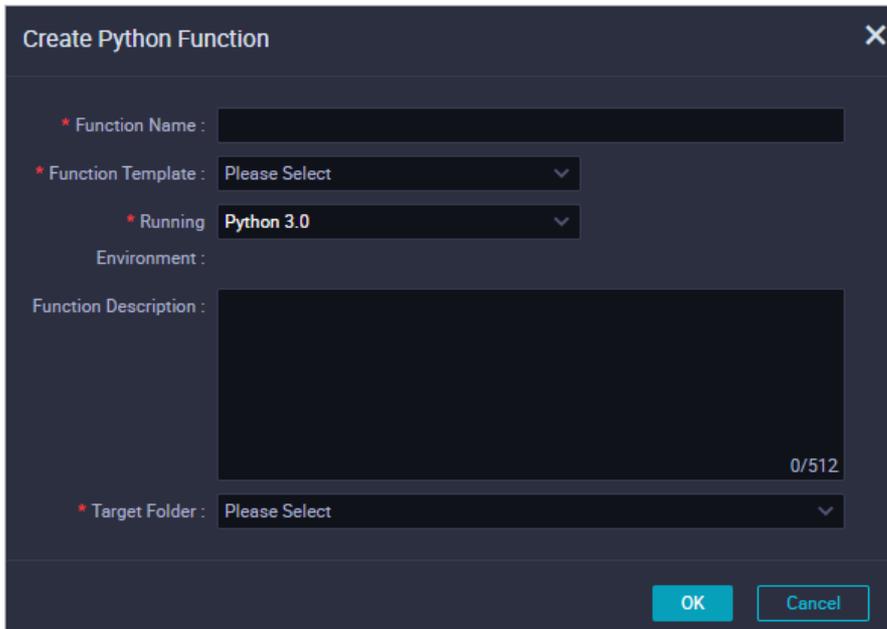
```
# -*- coding: utf-8 -*-
# event (str) : in filter it is the API result, in other cases, it is your param
# context : some environment information, temporarily useless
# import module limit: json,time,random,pickle,re,math
# do not modify function name
import json
def handler(event,context):
# load str to json object
obj= json.loads(event) # Convert the string specified by the event parameter to a JSON object.
# add your code here
# end add
return obj
```

You can modify the function template to write your own function. You can modify the names of the input parameters as needed.

Parameter 1 [context]: the context of calling APIs. The value is of the STRING type. This parameter is not in use and is left empty.
 Parameter 2 [event]: the result data returned by APIs or the preceding filter. The value is of the STRING type.

Create a Python function

1. Log on to the DataWorks console.
2. Click  in the upper-left corner and choose **All Products > Data Service**.
3. On the **Service Development** tab, move the pointer over  and choose **Function > Create Python Function**. You can also click a workflow, right-click **Function**, and then choose **New > Create Python Function**.
4. In the **Create Python Function** dialog box, set the parameters as required.



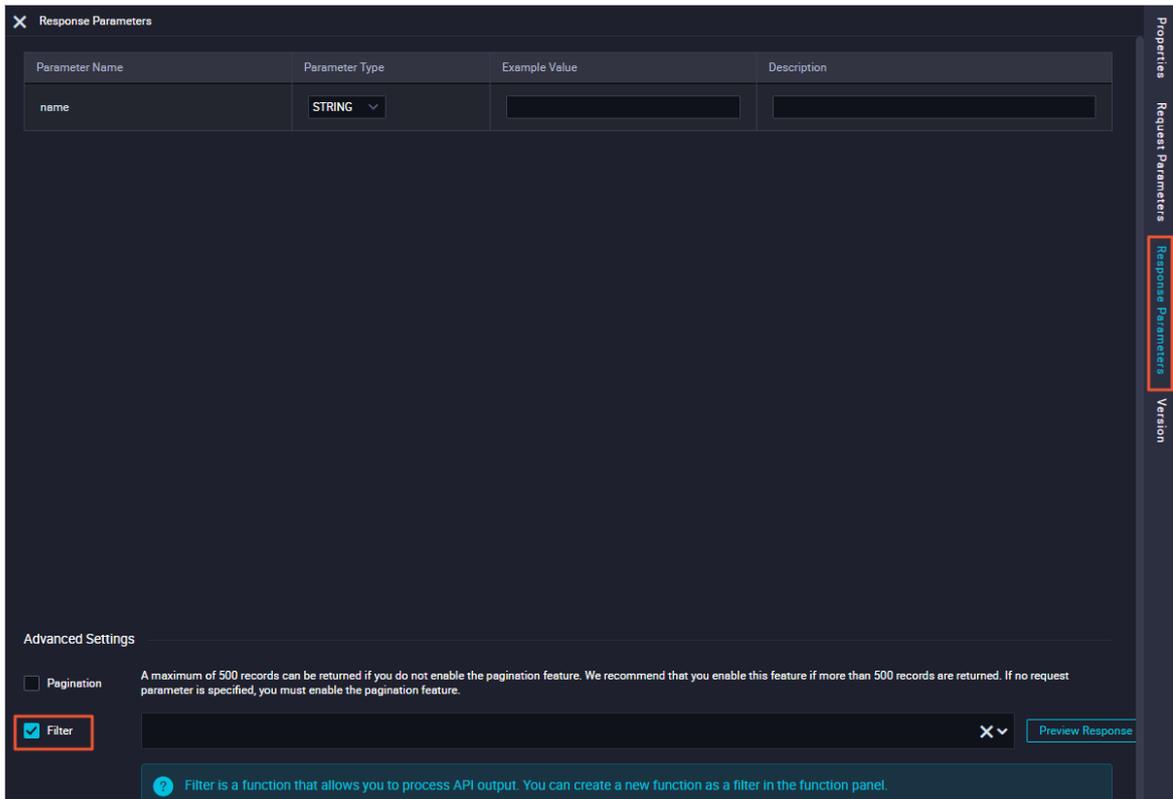
Parameter	Description
-----------	-------------

Parameter	Description
Function Name	The name of the function to create, which can be up to 256 characters in length.
Function Template	The template used to create the function. Set the value to Python3 Standard v1.
Running Environment	The runtime environment of the function. Set the value to Python 3.0.
Function Description	The description of the function.
Target Folder	The directory for storing the function.

5. Click **OK**.

Use post filters

1. On the **Service Development** tab, double-click the target API.
2. On the configuration tab that appears, click **Response Parameters** in the right-side navigation pane.
3. In the **Response Parameters** pane, select **Filter** in the **Advanced Settings** section.



4. Select functions from the **Filter** drop-down list.

Note A post filter is a function that is used to process the results returned by APIs. You can create a function and use it as a post filter.

5. Click **Preview Response** to view the processing results of the post filters.

2.14.6. Register APIs

This topic describes how to register APIs and manage and publish them to API Gateway together with APIs created based on data tables.

Currently, DataService Studio allows you to register only RESTful APIs. Supported request methods include GET, POST, PUT, and DELETE. Supported data types include forms, JSON data, and XML data.

Create a group

1. Log on to the DataWorks console.
2. Click the DataWorks icon in the upper-left corner and choose **All Products > DataService Studio**.
3. Go to the **Service Development** page, right-click **Service List**, and then select **New API Group**.
4. In the **Create API Group** dialog box that appears, enter values for **Group Name** and **Description**.
5. Click **OK**.

Configure basic API information

1. Right-click the new group and select **Register API**.
2. In the **Create API** dialog box that appears, set each parameter.

Configuration item	Description
API Name	The name must be 4 to 50 characters in length. It must start with a letter and can contain letters, digits, and underscores (_).
API Group	An API group is a collection of APIs for a specific feature or scenario. It is also the minimum API management unit of API Gateway. To create an API group, move the pointer over the Create icon and select New API Group .
API Path	API Path is the alias of Backend Service Path. APIs with different API paths can share the same backend service path and backend service host. Parameters defined in Backend Service Path must also be defined in brackets in API Path.
Protocol	Currently, HTTP and HTTPS are supported.
Request Method	You can select GET, POST, PUT, or DELETE as the request method. The parameters to be configured vary with the request method.
Return Type	Currently, JSON and XML return types are supported.
Description	The description of the API.

3. After configuring the basic API information, click **OK** to go to the API parameter configuration page.

Configure API parameters

On the API parameter configuration page, you need to define the backend service, request parameters, response content, and error codes.

Configuration item	Description
Define the back-end Service	<ul style="list-style-type: none"> • Backend Service Host: Enter the host of the API. The host must start with http:// or https://, and cannot contain the path. • Backend Service Path: Enter the path of the API. Place parameter names in brackets, for example, /user/[userid]. In the next step, parameters defined in Backend Service Path are automatically added to the request parameter list. • Backend timeout: Set the backend timeout period.
Define Request Parameters	<ul style="list-style-type: none"> • Parameter Type: The available request parameter locations (Path, Header, Query, or Body) vary with the request method. • Constant Parameters: A constant parameter is fixed and is invisible to the caller. You do not have to specify the constant parameters when calling an API. Instead, the constant parameters and their values are automatically sent to the backend service. This is useful when you want to set a parameter to a fixed value and hide the parameter value from the caller. • Request Body Definition: This configuration item is available only when the request method is POST or PUT. You can enter the body description in Request Body Definition. It is equivalent to an example of the request body so that API callers can refer to the format of the request body. The content type of the request body can be JSON or XML. <p> Note If a parameter has been defined in both the request body and the request parameter list, the parameter value in the request body takes priority.</p>
Define Response Content	You can enter a successful response example or an error response example for API callers to refer to when writing the return parse code.
Error Codes	<p>Enter the common errors and solutions in API calling. This enables API callers to troubleshoot and solve these errors.</p> <p> Note To ensure that the API is easily used by the callers, provide complete API parameter information if possible, especially the parameter sample values, default values, and sample responses.</p>

Test the API

After you have configured and saved the API parameters, click **Test** in the upper-right corner to go to the API test page.

Set the parameters and click **Test** to send an API request. The request and response details are displayed on the right. If the test fails, check carefully the error message, make modifications accordingly, and test the API again.

Pay attention to the settings of the successful response example in the configuration process. When testing an API, the system automatically generates error response examples and error codes. However, successful response examples are not automatically generated. To enable the system to save the test result as the successful response example, you need to select **Save as Successful Response Example** before performing the test. If the response contains sensitive data that must be masked, you can manually edit the response.

Note

- The successful response example is an important reference for API callers, and therefore must be configured.
- The Call Latency value is the latency of the current API request, which is used to evaluate the API performance. If the latency is long, consider optimizing the database.

After the API test is passed, return to the **Service Development** page, and click **Publish** to generate an API.

2.14.7. Test APIs

This topic describes how to test APIs.

When creating and registering an API, you can test the API. The system also provides an independent API test feature, which allows you to test APIs online.

1. Go to the **DataService Studio** page and click **Service Management** in the upper-right corner.
2. Click **Test API** in the left-side navigation pane.
3. Select the API to be tested, set the parameters, and then click **Test**.

Note

- The API Test page provides only online API testing. You cannot update or save the successful response example for an API on this page. To update the successful response example for an API, click the API name in the API list to enter the API edit mode. Then, update the successful response example for the API in the API testing step.
- You must test an API before publishing it.

2.14.8. Publish APIs

API Gateway provides API lifecycle management services, including API publishing, management, maintenance, and monetization. It helps you easily and quickly aggregate microservices, separate the front end from the backend, and integrate systems at low costs and low risks, making features and data available to partners and developers.

API Gateway provides permission management, traffic control, access control, and metering services. The services make it easy for you to create, monitor, and secure APIs. Therefore, we recommend that you publish the APIs that have been created and registered in DataService Studio to API Gateway. DataService Studio and API Gateway are interconnected, which allows you to publish APIs to API Gateway easily.

Publish APIs to API Gateway

Before publishing an API, you must activate API Gateway and register and test the API.

After the API passes the test, click **Publish** in the upper-right corner to publish the API to API Gateway.

The system automatically registers the API with API Gateway during the publish process. The system also creates a group in API Gateway with the same name as the API group to which the API belongs in DataService Studio and publishes the API in this group. After the API is published, you can access the API Gateway console to view API details or configure bandwidth throttling, access control, and other features.

If you generate an API to be called by your own application, you need to create an application in API Gateway, authorize the application to use the API, and enable the application to call the API by using AppKey and AppSecret. For more information, see *API Gateway documentation*. API Gateway also provides SDKs for mainstream programming languages to help you quickly integrate the API into your own application.

2.14.9. Call APIs

This topic describes how to call an API after this API is published on API Gateway.

API Gateway provides the SDKs for authorizing and calling APIs. You can authorize yourself, your associates, or third parties to use APIs. If you want to call an API, perform the following operations.

Three conditions for calling an API

The following three conditions must be met to call an API:

- **API:** The API that you call is clearly defined by API parameters.
- **App:** The app that you use to call the API has a key pair that uniquely identifies you. The key pair consists of the AppKey and AppSecret.
- **Relationship between the API and app:** If you want to call an API by using an app, the app must have the permission to call this API. This permission is granted through authorization.

Procedure

1. Obtain the API documentation.

The method of obtaining the API documentation varies depending on how you obtain the API. You can obtain the API by purchasing it from the marketplace, or you are authorized to use the API for free.

2. Create an app.

The app identifies you when you call the API. Each app has a key pair: AppKey and AppSecret, which are equivalent to an account and the corresponding password.

3. Obtain the permission to call the API.

Authorization means granting the app the permission to call an API. Your app must be authorized to call the API. The authorization method varies depending on how you obtain the API.

4. Call the API.

You can edit an HTTP or HTTPS request to call the API. Before calling the API, you can use examples of calling APIs in multiple languages in the API Gateway console to test the call.

You can call APIs after you passed the quick authentication or encrypted signature authentication.

Click **API call**. The AppCode and AppKey that are synchronized from API Gateway are available on the page that appears. You can perform **copy** and **reset** operations on them. For more information about API Gateway, see *API Gateway documentation*.

2.14.10. Use workflows

The workflow feature, also called service orchestration, provides a composite API service. This topic describes the benefits of workflows and how to use workflows.

In DataService Studio, a workflow is represented as a directed acyclic graph (DAG). By dragging and dropping nodes to a DAG, you can arrange APIs and functions in a serial, parallel, or branch structure based on the business logic.

When you run a workflow to call APIs, DataWorks runs the nodes in the workflow in sequence, passes parameters among the nodes, and automatically changes the status of each node. The workflow feature simplifies the process of calling multiple APIs or functions and reduces the cost of development and maintenance. This allows you to focus on business development.

Benefits

- Reduced cost of combining multiple APIs

By dragging and dropping nodes to a DAG, you can arrange APIs and functions in a serial, parallel, or branch structure without writing code. This reduces the cost of developing APIs.

- Higher performance in calling APIs and functions

A workflow allows you to call multiple APIs and functions in a container. Compared with writing code to call APIs and functions, the workflow feature reduces the latency of calling APIs and functions.

- Serverless architecture

The workflow feature adopts a serverless architecture. A serverless architecture supports automatic resource scaling based on business needs. You can focus on the business logic, without worrying about the runtime environment.

Obtain values of request and response parameters

DataService Studio uses JSONPath to obtain parameter values. JSONPath is a query language that allows you to extract data from JSON files.

For example, three nodes are run in the following order: A, B, and then C. Node C needs to use the response parameters of nodes A and B.

- Response parameter of node A: {"namea": "valuea"}

Expression for obtaining the value of the response parameter of node A: `${A.namea}`

- Response parameter of node B: {"nameb": "valueb"}

Expression for obtaining the value of the response parameter of node B: `$.nameb` or `${B.nameb}`

The built-in start node provides request parameters for the whole workflow. Assume that a request parameter of a workflow is {"namewf": "valuewf"}. All nodes of the workflow can obtain the value of the request parameter by using the `${START.namewf}` expression.

Set parameters

Request parameters:

- If you do not specify a value for a request parameter of a node, DataService Studio obtains the value of the same parameter in the first layer of the JSON string returned by the parent node, and assigns the value to the request parameter. If no value is specified for a request parameter of the first node, DataService Studio obtains the value of the same parameter in the request parameters of the workflow.
- If you specify a value for a request parameter, DataService Studio uses the value that you set.
- If you need to use the value of the specified parameter returned by a specified ancestor node, obtain the value by using a JSONPath expression.

Common JSONPath expressions for obtaining parameter values:

- `$.:` obtains response parameters of the parent node.
- `$.param:` obtains the value of the param parameter returned by the parent node. To allow you to obtain response parameters of any ancestor nodes, DataService Studio enhances JSONPath expressions.
- `${NODEID1}:` obtains response parameters of the node whose ID is NODEID1.
- `${START}:` obtains request parameters of the workflow, which are response parameters of the start node.
- `${NODEID1.param}:` obtains the value of the param parameter returned by the node whose ID is NODEID1.

JSONPath expressions for setting response parameters of a node:

- `$.:` sets response parameters of the current node.
- `$.param:` sets the param parameter to be returned by the current node.
- `${NODEID1.param}:` sets the param parameter returned by the node whose ID is NODEID1.

Example

Add a connection before you create and use a workflow. In this example, a MySQL connection is used.

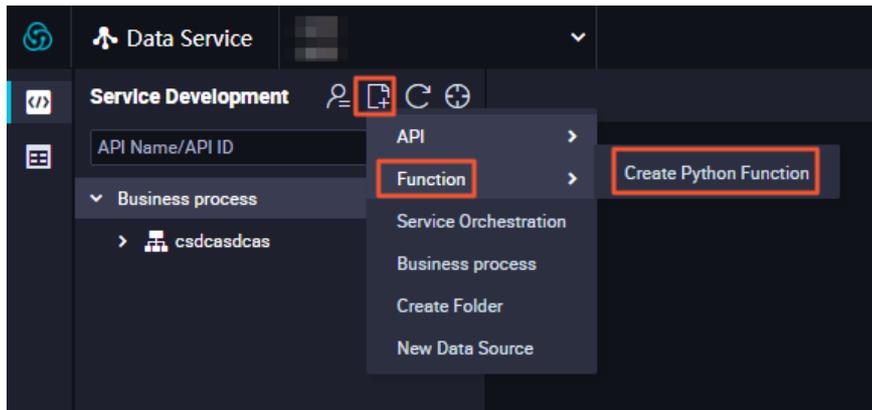
1. Register an API.

In this example, create an API by using the registration method. For more information, see [Register an API](#).

2. Register a function. For more information, see [Create a function](#).

In this example, create a Python function as a branch node to process the result data of the parent node.

- i. Go to the **Service Development** tab, move the pointer over  and choose **Function > Create Python Function**.



- ii. In the **Create Python Function** dialog box, set the parameters as required.

Parameter	Description
Function Name	The name of the function to create, which can be up to 256 characters in length.
Function Template	The template used to create the function. Set the value to Python3 Standard v1.
Running Environment	The runtime environment of the function. Set the value to Python 3.0.
Function Description	The description of the function. The description can be up to 512 characters in length.
Target Folder	The directory for storing the function.

- iii. Click **OK**.
- iv. On the configuration tab of the function, configure the function.
- a. In the **Edit Code** section, enter the function code.

```
# -*- coding: utf-8 -*-
# event (str) : in filter it is the API result, in other cases, it is your param
# context : some environment information, temporarily useless
# import module limit: json,time,random,pickle,re,math
import json
def handler(event,context):
    # load str to json object
    obj = json.loads(event)
    # add your code here
    # end add
    return obj
```

- b. In the **Environment Configuration** section, set the **Memory** and **Function Timeout** parameters.

- v. Click Save icon in the toolbar.
3. Create a workflow.
- i. On the **Service Development** tab, move the pointer over  and select **Service Orchestration**.
 - ii. In the **Service Orchestration** dialog box, set the parameters as required.

Parameter	Description
API Name	The name of the workflow. The name must be 4 to 50 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.
API Path	The path for storing the workflow, for example, <i>/user</i> .
Call Mode	The mode for calling the API to be arranged in the workflow. Valid values: Synchronous Call and Asynchronous Call . <ul style="list-style-type: none"> ▪ If you set this parameter to Synchronous Mode, the API returns results immediately after it is called. The synchronous mode is most commonly used. ▪ If you set this parameter to Asynchronous Mode, the API returns the RequestID parameter immediately after it is called. The API caller can then obtain the call result from a message queue based on the request ID.
Protocol	The protocol used by the API. Valid values: HTTP and HTTPS .

Parameter	Description
Request Method	The request method used by the API. Valid values: GET and POST .
Response Content Type	The return type of the API. Set the value to JSON .
Visible Range	<p>The visibility of the workflow. Valid values:</p> <ul style="list-style-type: none"> ▪ Work Space: The workflow is visible to all members in the current workspace. ▪ Private: The workflow is visible only to its owner and permissions on the workflow cannot be granted to other users. <p>Note If you set the Visible Range parameter to Private, the workflow is visible only to you in the directory tree. It is hidden to other members of the workspace.</p>
Label	<p>The tag of the workflow. Select one or more tags from the drop-down list. For more information, see Manage tags.</p> <p>Note You can set at most five tags for a workflow.</p>
Description	The description of the workflow, which can be up to 2,000 characters in length.
Target Folder	The directory for storing the workflow.

iii. Click **OK**.

4. Configure the workflow.

- i. On the configuration tab of the workflow, drag and drop nodes and connect them as required.
- ii. Double-click the **API1** node to edit the node. Select the API that you registered earlier as the API to be called in the node.

Select **set output results** and enter `{"user_id": "$.data[0].id"}`.

Use JSONPath expressions to set response parameters. The syntax for obtaining the value of a parameter is `$(NodeA.namea)`, which is the same as that for setting request parameters. `{"user_id": "$.data[0].id"}` assigns the value of the id parameter of the first element in the data array to the user_id parameter. Then, the API1 node returns `{"user_id": "value"}` in JSON format.

- iii. Double-click the **PYTHON1** node to edit the node. Select the function that you created earlier as the function to be called in the node.

- iv. Double-click the **SWITCH2** node to edit the node. In the right-side pane that appears, click **Set branch conditions**. In the Set branch conditions dialog box, enter conditional expressions based on the response parameter of the parent node. For example, you can enter expressions in the `{Node ID}.Parameter>1` or `$.Parameter>1` format. Conditional expressions support the following operators: `==`, `!=`, `>=`, `>`, `<=`, `<`, `&&`, `!`, `()`, `+`, `-`, `*`, `/`, and `%`.

In this example, the `user_id` parameter is the response parameter of the API1 node and is used as the request parameter of the SWITCH2 node.

Branch node 1: `$.user_id != 1`, indicating that the branch node 1 is run if the value of the `user_id` parameter is not 1.
Branch node 2: `$.user_id == 1`, indicating that the branch node 2 is run if the value of the `user_id` parameter is 1.

- v. Double-click the end node and then click the **Response Parameters** tab on the right side to set response parameters.
5. Click **Test** in the upper-right corner.
 6. In the **Test APIs** dialog box, set the parameters as required and click **OK**.

You can view the test result after the workflow is tested.

2.14.11. Manage versions

This topic describes how to manage versions of APIs, workflows, and functions in Data Service.

You can view and compare historical versions of APIs, workflows, and functions. Data Service generates a version record each time an API, a workflow, or a function is published.

View historical versions

1. Log on to the DataWorks console. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > DataService Studio**.
2. On the **Service Development** page that appears, double-click the name of the target API in the API list.

 **Note** You can also click **Function** or **Table** in the left-side navigation pane and double-click the name of a workflow or function to manage its versions.

3. In the right-side navigation pane, click **Version**. In the **Version** dialog box that appears, view historical versions of the API.

Parameter	Description
API ID	The ID of the API. Each API ID is unique.
Version	The version of the API. A version is generated each time the API is published. V1 indicates version 1 and V2 indicates version 2. The version number is incremented by 1 each time.
The Author	The user who published the version.
Submitted Date	The time when the version was published. The time is accurate to second.

Parameter	Description
Status	<p>The status of the version. Value values:</p> <ul style="list-style-type: none"> ◦ Release: indicates that the version of the API is the latest version. ◦ Off-Line: indicates that the version of the API is a historical version.
Actions	<p>The operations that you can perform on the version. The Actions column is only available for API and workflow versions. You can click API Details of a version to go to the details page of the version. The Actions column is unavailable for function versions.</p>

Compare historical versions

In the **Version** dialog box, select two versions to compare, and click **Contrast**. In the **History Version Contrast** dialog box that appears, compare the code and parameters of the two versions.

2.14.12. FAQ

This topic provides answers to commonly asked questions about DataService Studio.

- Q: Do I need to activate the API Gateway service?

A: API Gateway provides you with high-performance and highly available API hosting services. If you need to make your APIs available to others, activate the API Gateway service first.

- Q: Where can I add and change connections?

A: After you log on to the DataWorks console, click the DataWorks icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page. In the left-side navigation pane, choose **Sync Resources > Connections**. On the page that appears, perform the relevant configuration. DataService Studio automatically reads data from the connections that you have configured.

- Q: What is the role of an API group in DataService Studio? What is the relationship between an API group in DataService Studio and an API group in API Gateway?

A: An API group is a set of APIs specific to a feature or scenario. It is the smallest organization unit in DataService Studio, which is similar to an API group in API Gateway. An API group in DataService Studio is equivalent to an API group in API Gateway. After you publish an API from DataService Studio to API Gateway, API Gateway automatically creates an API group with the same name.

- Q: How can I configure an API group appropriately?

A: Typically, an API group includes APIs that provide similar features or resolve a specific issue. For example, a weather API group can include APIs that are used to check the weather by city and by longitude and latitude.

- Q: How many API groups can I create?

A: An Alibaba Cloud account can create up to 100 API groups.

- Q: When do I need to enable the pagination feature for an API call so that its return results can be displayed on multiple pages?

A: By default, an API call returns a maximum of 2,000 records. If an API call may return more than 2,000 records, enable the pagination feature. If you do not specify any request parameters, the API call usually returns a large number of records and the pagination feature is automatically enabled.

- Q: Do APIs created by DataService Studio support POST requests?

A: APIs created by DataService Studio support GET and POST requests.

- Q: Do APIs created by DataService Studio support the HTTPS protocol?

A: APIs created by DataService Studio support both HTTP and HTTPS protocols.

2.15. Stream Studio

2.15.1. Overview

Built on Alibaba Cloud Realtime Compute, which is based on Apache Flink, Stream Studio allows you to develop real-time computing nodes in directed acyclic graph (DAG) mode or SQL mode. You can switch between the two modes to edit the code or drag and drop components and configure them in a visual way.

As an ideal platform for developing real-time computing nodes, Stream Studio has the following features:

- Supports developing nodes in DAG mode. You can perform drag and drop components to configure real-time computing nodes.
- Supports developing nodes in SQL mode. You can edit the code to configure real-time computing nodes.
- Supports switching between the DAG mode and the SQL mode for you to easily check SQL operators.
- Supports using Function Studio to create and publish user-defined functions (UDFs) online in exclusive mode.
- Supports smart diagnosis for real-time computing nodes to facilitate online troubleshooting.

2.15.2. Bind a Realtime Compute project

1. Log on to the DataWorks console.
2. Click the **Project Manage** icon in the upper-right corner. The **Project Management** page appears.
3. On the **Project Management** page, click **Add Compute Engine** in the **Compute Engine** section, and select **Add engine service**.
4. Enter the name of the Blink engine to be added and click **Bind**.

2.15.3. Create a real-time computing node

This topic describes how to create a real-time computing node and develop data in Stream Studio.

Prerequisites

A workflow is created. You can create real-time computing nodes and develop data under an existing workflow.

Procedure

After a workflow is created, you can create real-time computing nodes under the workflow. By default, data is developed for a real-time computing node in directed acyclic graph (DAG) mode.

1. Right-click the workflow that you have created and select **Create task**.
2. In the **Create Node** dialog box that appears, set the parameters and click **Submit**.
3. Develop data in DAG mode on the **Components** page.

The Components page includes the following four sections:

- **Component list** section: In this section, you can view the list of available components. You can click **Components** on the left-side navigation submenu to go to the Components page and view the list.
- **DAG** section: In this section, you can drag and drop components to the DAG and connect them. To configure the dependency between two components, click and hold the highlighted dot at the

bottom of a component and move the pointer to link this component with a descendant component. A DAG corresponds to a real-time computing node.

- Parameter configuration section: Double-click a component in the DAG. Then, you can set the related parameters in this section.
- Toolbar section: In this section, you can click the icons to perform the save, submit, steal lock, pre-compilation, test, stop, reload, and format operations respectively.

When you configure the DAG, you can right-click a component and select an operation from the menu that appears to perform it on the selected component. Available operations include **Rename**, **View schema**, **Delete node**, **View error message**, **New component group**, and **Copy**.

2.15.4. Get started with Stream Studio

This topic uses an example to describe how to use Stream Studio to develop and manage a real-time computing node. Stream Studio allows you to create, configure, publish, run, stop, and unpublish a real-time computing node.

Prerequisites

A Realtime Compute project is bound to the current DataWorks workspace.

Context

- Data store: a Datahub table with a created topic, which contains the `m_name`, `id`, `m_type`, and `tag` fields.

 **Note** The Datahub topic must be created in advance.

- Data processing: splits the tag field by using the `semicolon (;)` as the delimiter to the color, mode, and weight fields.
- Output data: writes the `id`, `m_type`, and `weight` fields to a Log Service table.

 **Note** The Log Service project and Logstore must be created in advance.

Procedure

1. Log on to the DataWorks console. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Stream Studio**.
2. Create a workflow.
 - i. On the Stream Studio homepage, click **New business process**.
You can also move the pointer over the **Create** icon and click **Business process**.
 - ii. In the **Create business process** dialog box that appears, set the **Business Name** and **Description** parameters.
 - iii. Click **New**.
3. Create a real-time computing node.
 - i. Right-click the workflow that you have created and select **Create task**.
 - ii. In the **Create node** dialog box that appears, set the relevant parameters.
 - iii. Click **Submit**. The **Components** page appears.

- iv. On the left-side navigation submenu, click **Resource Reference** and select **PUBLIC_COMMON**.

 **Note** If this option is not selected, the message shown when you use the `FixedFieldsSplit` component.

You can also go to the Resource Reference page to configure the reference resources after this message appears.

- 4. Configure the created real-time computing node.

On the left-side navigation submenu, click **Components**. The **Components** page appears.

- i. Configure the data store of the node on the **Components** page.
 - a. Drag and drop the Datahub component in the Data Source section to the directed acyclic graph (DAG).
 - b. Click the Datahub component and set the parameters as needed on the **Parameter configuration** tab that appears.

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name. If this parameter is not specified, the real table name is used.
table schema	The custom fields and attribute fields to be returned. Click Custom . In the Select field dialog box that appears, click + Add and enter the name and type of the output field. Then, click OK .
endPoint	The endpoint used to access Datahub. It corresponds to the <code>endPoint</code> parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID used to read data from Datahub. It corresponds to the <code>accessId</code> parameter in the WITH clause of the CREATE TABLE statement.
accessKey	The AccessKey secret used to read data from Datahub. It corresponds to the <code>accessKey</code> parameter in the WITH clause of the CREATE TABLE statement.
project	The name of the Datahub project from which data is to be read. It corresponds to the <code>project</code> parameter in the WITH clause of the CREATE TABLE statement.
topic	The name of the Datahub topic from which data is to be read. It corresponds to the <code>topic</code> parameter in the WITH clause of the CREATE TABLE statement.
startTime	The beginning of the time range when data is read. It corresponds to the <code>startTime</code> parameter in the WITH clause of the CREATE TABLE statement.
maxRetryTimes	The maximum number of retries for reading data from Datahub. It corresponds to the <code>maxRetryTimes</code> parameter in the WITH clause of the CREATE TABLE statement. Default value: 20.

Parameter	Description
retryIntervalMs	The retry interval at which data is read. It corresponds to the <code>retryIntervalMs</code> parameter in the <code>WITH</code> clause of the <code>CREATE TABLE</code> statement. Unit: milliseconds. Default value: <code>1,000</code> .
batchReadSize	The number of data records that are read at a time. It corresponds to the <code>batchReadSize</code> parameter in the <code>WITH</code> clause of the <code>CREATE TABLE</code> statement. Default value: <code>10</code> .
lengthCheck	The rule for checking the number of fields parsed from a row of data. It corresponds to the <code>lengthCheck</code> parameter in the <code>WITH</code> clause of the <code>CREATE TABLE</code> statement. Default value: <code>NONE</code> .
columnErrorDebug	Specifies whether to enable debugging. It corresponds to the <code>columnErrorDebug</code> parameter in the <code>WITH</code> clause of the <code>CREATE TABLE</code> statement. If you turn on this switch, logs about parsing errors are returned. You can view the node details Operation Center.

ii. Configure the data operator.

- a. Drag and drop the **FixedFieldsSplit** component to the DAG to split the tag field.
- b. Click and hold the highlighted dot at the bottom of the **Datahub** component and move the pointer to link this component with the **FixedFieldsSplit** component.
- c. Click the **FixedFieldsSplit** component and set the field to tag and the column separator to semicolon (;) on the **Parameter configuration** tab that appears.
- d. Click **Custom** for the Add column parameter. In the **Select field** dialog box that appears, click **+** Add and enter the name and type of the output field. Then, click **OK**.
- e. Drag and drop the **Select** component to the DAG. Click and hold the highlighted dot at the bottom of the **FixedFieldsSplit** component and move the pointer to link this component with the **Select** component.
- f. Click the **Select** component and click **0 Field has been selected** on the **Parameter configuration** tab that appears.
- g. In the dialog box that appears, select the fields to be returned and click **OK**.

iii. Configure the result table.

This example uses the **LogService** component as the destination.

- a. Drag and drop the **LogService** component to the DAG. Click and hold the highlighted dot at the bottom of the **Select** component and move the pointer to link this component with the **LogService** component.

- b. Click the **LogService** component and set the parameters as needed on the **Parameter configuration** tab that appears.

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name. If this parameter is not specified, the real table name is used.
Output Field	The fields to be returned. Click 0 Field has been selected for the Output Field parameter. In the dialog box that appears, select the fields to be returned and click OK .
endPoint	The endpoint used to access Log Service. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
project	The name of the Log Service project to which data is to be written. It corresponds to the project parameter in the WITH clause of the CREATE TABLE statement.
topic	The name of the Log Service topic to which data is to be written. It corresponds to the topic parameter in the WITH clause of the CREATE TABLE statement.
source	The name of the Log Service table to which data is to be written. It corresponds to the source parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID used to access Log Service. It corresponds to the accessId parameter in the WITH clause of the CREATE TABLE statement.
accessKey	The AccessKey secret used to access Log Service. It corresponds to the accessKey parameter in the WITH clause of the CREATE TABLE statement.
mode	The mode of data writing. It corresponds to the mode parameter in the WITH clause of the CREATE TABLE statement. Default value: <i>random</i> .
logStore	The name of the Logstore in the Log Service project to which the data is to be written. It corresponds to the logStore parameter in the WITH clause of the CREATE TABLE statement.

- iv. Switch between the DAG mode and SQL mode.

Stream Studio allows you to configure a real-time computing node in both DAG mode and SQL mode. You can switch between these two modes.

By default, you configure a node in DAG mode. You can click **Switch to SQL mode** in the upper-right corner to switch to the SQL mode.

In SQL mode, you can click **Switch to DAG mode** in the upper-right corner to switch back to the DAG mode.

- v. Configure the execution plan.

- a. Click **Execution Plan** on the right-side navigation submenu to generate an execution plan.
- b. Click **Save execution plan**.

5. Publish the real-time computing node.

You can publish the real-time computing node that you have configured. Click **Save** and then click **Submit** to publish the node.

- i. Click **Save** and then click **Submit**. If you have not saved the node, a message appears, indicating that you must save it.
- ii. In the **Submit New version** dialog box that appears, enter the remarks for the node and click **OK**.
- iii. After you publish the node, you can go to the **OAM** page to view the node status and manage the node.

6. Perform O&M on the real-time computing node.

Click **OAM** in the upper-right corner to perform O&M on the real-time computing node.

i. Start the real-time computing node.

Find the real-time computing node that you have created in the node list and click **Start** to start the node.

You can set a custom start time for the real-time computing node based on your business requirement.

After starting the real-time computing node, you can click the node name to view its running status. If the real-time computing node is started properly, it enters the **Run** state.

ii. Stop and unpublish the real-time computing node.

- a. Click **Stop** to stop the real-time computing node.
- b. After the real-time computing node is stopped, click **Offline** to unpublish it.

Result

Now you have created, configured, published, run, stopped, and unpublished a real-time streaming node.

2.15.5. Configure components

2.15.5.1. Source tables

2.15.5.1.1. Datahub

Datahub is a real-time data distribution platform that is designed to process streaming data. It provides a channel for the Apsara Stack DTplus platform to process big data.

Realtime Compute typically uses Datahub to store source and result tables for streaming data processing. Data Transmission Services (DTS) and the Internet of Things (IoT) also use Datahub to access big data platforms. Datahub stores streaming data that can be used as input data for Realtime Compute.

Parameter configuration

Parameter	Description	Remarks
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.	None
table schema	The custom fields and attribute fields to be read from Datahub.	None
endPoint	The consumer endpoint.	None

Parameter	Description	Remarks
accessId	The AccessKey ID used to read data from Datahub.	None
accessKey	The AccessKey secret used to read data from Datahub.	None
project	The name of the Datahub project from which data is to be read.	None
topic	The name of the Datahub topic from which data is to be read.	None
startTime	The beginning of the time range when data is read.	The format is <code>yyyy-MM-dd hh:mm:ss</code> .
maxRetryTimes	The maximum number of retries for reading data from Datahub.	None
retryIntervalMs	The retry interval at which data is read. Unit: milliseconds.	None
batchReadSize	The number of data records that are read at a time.	None
lengthCheck	The rule for checking the number of fields parsed from a row of data.	<p>Valid values: <i>SKIP</i>, <i>EXCEPTION</i>, and <i>PAD</i>. Default value: <i>SKIP</i>.</p> <ul style="list-style-type: none"> • <i>SKIP</i>: skips a data record when the number of fields in the data record is not the specified one. • <i>EXCEPTION</i>: throws an exception when the number of fields in the data record is not the specified one. • <i>PAD</i>: pads fields in sequence. Pad a field with null when the field does not exist.
columnErrorDebug	Specifies whether to enable debugging. If you turn on this switch, logs about parsing errors are returned.	None
BLOB	Specifies whether the type of data read from Datahub is BLOB.	None
Data Quality	Specifies whether to open the Data Quality page to view related monitoring nodes.	None

Field type mapping

The following table lists the mapping between Datahub and Realtime Compute data types. We recommend that you declare the type mapping in the DDL statement.

Datahub data type	Realtime Compute data type
BIGINT	BIGINT
DOUBLE	DOUBLE
TIMESTAMP	BIGINT
BOOLEAN	BOOLEAN
DECIMAL	DECIMAL

Attribute fields

You can obtain the attribute field indicating the system time at which each data record is written to Datahub.

Field	Description
System Time	The system time at which each data record is written to Datahub.

2.15.5.1.2. Log Service

As an all-in-one real-time data logging service, Log Service allows you to quickly finish tasks such as data ingestion, consumption, delivery, query, and analysis without any extra development work. This can help you improve O&M and operational efficiency, and build up the capability to process large amounts of logs in the data technology era.

Log Service stores streaming data that can be used as input data for Realtime Compute.

The data format of Log Service is consistent with JSON. Example:

```
{
  "a": 1000,
  "b": 1234,
  "c": "li"
}
```

Parameter configuration

Parameter	Description	Remarks
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.	None
table schema	The custom fields and attribute fields to be read from Log Service.	None
endPoint	The consumer endpoint.	Log Service endpoints
accessId	The AccessKey ID used to read data from Log Service.	None
accessKey	The AccessKey secret used to read data from Log Service.	None

Parameter	Description	Remarks
project	The name of the Log Service project from which data is to be read.	None
logStore	The name of the Logstore under the Log Service project.	None
consumerGroup	The name of the consumer group.	You can specify a custom consumer group name. The format of the name is not fixed.
startTime	The beginning of the time range when the log data is consumed.	None
heartBeatIntervalMills	Optional. The heartbeat interval at which the client sends heartbeat messages. Unit: milliseconds.	None
maxRetryTimes	The maximum number of retries for reading data from Log Service.	None
columnErrorDebug	Specifies whether to enable debugging. If you turn on this switch, logs about parsing errors are returned.	None

Field type mapping

The following table lists the mapping between Log Service and Realtime Compute data types. We recommend that you declare the type mapping in the DDL statement.

Log Service data type	Realtime Compute data type
STRING	VARCHAR

Attribute fields

Currently, Log Service supports the following three attribute fields by default. You can also specify other custom fields.

Field	Description
<code>__source__</code>	Specifies a log source.
<code>__topic__</code>	Specifies a log topic.
<code>__timestamp__</code>	Specifies the time when a logged event occurs.

Note

- Currently, Log Service does not support the MAP type.
- We recommend that you define the fields in the same order as the fields in the preceding table. Unordered fields are also supported.
- If the input data is in JSON format, define the delimiter and use the built-in function JSON_VALUE to parse the JSON value. Otherwise, the parsing fails and the following error is returned:

```
2017-12-25 15:24:43,467 WARN [Topology-0 (1/1)] com.alibaba.blink.streaming.connectors.common.source.parse.DefaultSourceCollector - Field missing error, table column number: 3, data column number: 3, data field number: 1, data: [{"lg_order_code":"LP00000005","activity_code":"TEST_CODE1","occur_time":"2017-12-10 00:00:01"}]
```

- The batchSize value must not exceed 1,000. Otherwise, an error occurs.
- The batchSize parameter specifies the number of log items read at a time in a log group. If both the size of a single log item and the batchSize value are too large, frequent garbage collection (GC) may be triggered. To avoid this, you must set batchSize parameter to a smaller value.

2.15.5.2. Dimension tables

2.15.5.2.1. ApsaraDB RDS

ApsaraDB RDS is a stable, reliable, and scalable cloud database service.

Parameter configuration

Parameter	Description	Remarks
-----------	-------------	---------

Parameter	Description	Remarks
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.	None
url	The URL of your ApsaraDB RDS instance.	None
tableName	The name of the table in your database.	None
userName	The username that is used to connect to the database.	None
password	The password that is used to connect to the database.	None
Output Field	The fields that you want to return to the descendant component.	None
maxRetryTimes	The maximum number of retries for reading data from the table.	None
Cache Policy	The policy that is used to cache data.	Valid values: <i>None</i> , <i>LRU</i> , and <i>ALL</i> .
primaryKey	The primary key field in the output fields.	<ul style="list-style-type: none"> You must specify a primary key when you declare a dimension table. When you join a dimension table with another table, the ON clause must contain the equivalent (=) conditions for all the primary key fields. The primary key in ApsaraDB RDS or PolarDB-X is the primary key or unique index column of an ApsaraDB RDS or PolarDB-X dimension table.

Additional information

- ApsaraDB RDS and PolarDB-X provide the following cache policies:

- None:** indicates that data is not cached.
- LRU:** indicates that only the recently used data is cached.

If this cache policy is selected, you must specify the `cacheSize` and `cacheTTLms` parameters.

- ALL:** indicates that all data is cached.

Before the system runs a node, it loads all the data in the remote table to the memory. Then, the system searches the cache for data in all dimension table queries. If a cache miss occurs, all data is cached again after the cache times out. The ALL cache policy applies to scenarios where the remote table is small but a large number of missing keys exist. If this cache policy is selected, you must specify the `cacheTTLms` and `cacheReloadTimeBlackList` parameters.

- If the ALL cache policy is used, the system reloads data asynchronously. Therefore, you must increase the memory of the JOIN operator. The size of the increased memory is twice the data size of the remote table.
- If the ALL cache policy is used, pay special attention to the memory of the JOIN operator to prevent out of memory (OOM) errors.

2.15.5.2.2. Tablestore

Tablestore is a distributed NoSQL database service that is developed based on the Apsara distributed operating system. It features high availability and data reliability.

Parameter configuration

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.
instanceName	The name of the instance.
tableName	The name of the table.
Output Field	The fields that you want to return to the descendant component.
endPoint	The endpoint of the instance. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID of the account that you use to read data from the table.
accessKey	The AccessKey secret of the account that you use to read data from the table.

Parameter	Description
Cache Policy	The policy used to cache data. Valid values: None and LRU .
primaryKey	The primary key field in the output fields.

2.15.5.2.3. MaxCompute

This topic describes the parameter configuration, field type mapping, and metrics of a MaxCompute dimension table.

Parameter configuration

Parameter	Description	Remarks
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.	None
endPoint	The endpoint used to access MaxCompute. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.	None
tunnelEndPoint	The endpoint of the Tunnel service. It corresponds to the tunnelEndPoint parameter in the WITH clause of the CREATE TABLE statement.	This parameter is required for a MaxCompute dimension table deployed in a Virtual Private Cloud (VPC).
project	The name of the MaxCompute project to which the dimension table belongs.	None
accessId	The AccessKey ID used to read data from MaxCompute.	None
accessKey	The AccessKey secret used to read data from MaxCompute.	None
Output Field	The fields to be returned to the descendant component.	None
partition	The partition name of the MaxCompute dimension table.	None
maxRowCount	The maximum number of data records that can be read from the MaxCompute dimension table	None
Cache Policy	The policy for caching data.	Default value: ALL .

Parameter	Description	Remarks
cacheSize	The maximum number of data records that can be cached.	This parameter is required if you set the Cache Policy parameter to LRU . Default value: 100,000.
cacheTTLms	The time interval at which the cache is refreshed. It corresponds to the cacheTTLms parameter in the WITH clause of the CREATE TABLE statement. Units: milliseconds.	This parameter specifies the cache refresh interval when the Cache Policy parameter is set to ALL . The cache is not refreshed by default.
cacheReloadTimeBlackList	Optional. The time period during which the cache is not refreshed. This parameter is valid when the Cache Policy parameter is set to ALL . During the time period specified by this parameter, for example, the Double 11 Shopping Festival, the cache is not refreshed.	This parameter is left empty by default. If you want to set this parameter, specify the time period in the format shown in the following example: 2017-10-24 14:00 -> 2017-10-24 15:00, 2017-11-10 23:30 -> 2017-11-11 08:00 Separate multiple time periods with commas (.). Separate the start and end time for a time period with the string "->".
primaryKey	The primary key field of the output fields.	None

Field type mapping

MaxCompute data type	Realtime Compute data type
TINYINT	TINYINT
SMALLINT	SMALLINT
INT	INT
BIGINT	BIGINT
FLOAT	FLOAT
DOUBLE	DOUBLE
BOOLEAN	BOOLEAN
DATETIME	TIMESTAMP
TIMESTAMP	TIMESTAMP
VARCHAR	VARCHAR
STRING	STRING
DECIMAL	DECIMAL

MaxCompute data type	Realtime Compute data type
BINARY	VARBINARY

Metrics

When you join the dimension table to another table, you can view metrics such as the correlation degree and cache hit ratio. You can use K-Monitor to view the metrics.

Query statement	Description
fetch qps	Queries the total number of queries per second (QPS) against the dimension table, including hits and misses. The metric name is <code>blink.projectName.jobName.dimJoin.fetchQPS</code> .
fetchHitQPS	Queries the number of hits (in QPS) against the dimension table, including cache hits and hits against the physical dimension table. The metric name is <code>blink.projectName.jobName.dimJoin.fetchHitQPS</code> .
cacheHitQPS	Queries the number of cache hits (in OPS) against the dimension table. The metric name is <code>blink.projectName.jobName.dimJoin.cacheHitQPS</code> .
dimJoin.fetchHit	Queries the correlation degree of the dimension table and the table to which the dimension table is joined. The metric name is <code>blink.projectName.jobName.dimJoin.fetchHit</code> .
dimJoin.cacheHit	Queries the cache hit ratio of the dimension table. The metric name is <code>blink.projectName.jobName.dimJoin.cacheHit</code> .

Note

- We recommend that you use Realtime Compute V2.1.1 and later.
- To use a MaxCompute dimension table, you must grant the read permission to the account for accessing MaxCompute.
- When you declare a dimension table, you must specify the primary key. When you join a dimension table with another table, the ON condition must contain an equivalent condition that includes the primary key of either table.
- The primary key value for each row of a MaxCompute dimension table must be unique. Otherwise, the duplicate records are removed.
- If the dimension table is a partitioned table, Realtime Compute does not currently support writing the partition key column to the schema.
- When the cache policy is set to ALL, Realtime Compute reloads data asynchronously. Therefore, you must increase the memory of the JOIN operator. The size of the increased memory is twice the data size of the remote table.
- The following failover message may appear when you run a node:

```
RejectedExecutionException: Task
java.util.concurrent.ScheduledThreadPoolExecutor$ScheduledFutureTask,
```

Generally, this message appears because dimension table joining in Realtime Compute V1.x has certain issues. We recommend that you upgrade Realtime Compute to V2.1.1 or later. If you want to continue using the existing version, we recommend that you pause the node and resume it after troubleshooting. To troubleshoot the failover, check the specific error information that was generated for the first failover record in the failover history.

2.15.5.3. Data operators

2.15.5.3.1. Filter

The Filter component allows you to configure filter conditions. It corresponds to the WHERE clause in SQL statements.

Parameter configuration

Enter the filter expression to configure this component. The filter expression supports functions and operators (=, <>, >, >=, <, and <=), for example, `city = 'Beijing'`.

2.15.5.3.2. GroupBy

The GroupBy component corresponds to the GROUP BY clause in SQL statements.

Parameter configuration

Parameter	Description
Select grouping field	The fields based on which data is grouped. You can specify multiple fields.
Output Field	The fields to be returned, that is, the fields to be selected. You can specify the fields in the same way that you configure the Select component.

2.15.5.3.3. Join

The Join component corresponds to the JOIN clause in SQL statements.

Parameter configuration

Parameter	Description
JoinMode	The JOIN mode to be used. Valid values: INNER JOIN, LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN.
expression	The JOIN expression. An equijoin is supported, for example, <code>leftId = rightId AND limit = 0</code> , whereas a non-equijoin is not supported.
Select Field	The fields to be returned, that is, the fields to be selected.

2.15.5.3.4. Select

The Select component allows you to configure the fields to be returned and supports field expressions. It corresponds to SELECT statements.

Parameter configuration

Select or configure the output fields in the Select field dialog box.

You can select fields to be returned in the **Field list** section and set an alias for a field in the **Field alias** column. To set a field expression, click the Edit icon next to the target field name. In the Edit dialog box that appears, enter the required SQL statement.

2.15.5.3.5. UDTF

The UDTF component allows you to configure custom functions. It corresponds to the UDTF clause in SQL statements.

Parameter configuration

Parameter	Description
JoinMode	The JOIN mode for the custom function. Only <i>INNER JOIN</i> and <i>LEFT OUTER JOIN</i> are supported. <ul style="list-style-type: none">• <i>INNER JOIN</i>: returns an empty result set when the UDTF clause returns no result.• <i>LEFT OUTER JOIN</i>: returns the NULL string when the UDTF clause returns no result.
Select function	The name of the function that the current node references. To reference a function for the current node, upload the related resources on the Resource Reference page and select the target resource.
parameter expression	The input parameters and output parameters of the referenced function.
Output Field	The fields to be returned. You can configure the name, alias, and expression of each field.

2.15.5.3.6. UnionAll

The UnionAll component corresponds to the UNION ALL clause in SQL statements.

Parameter configuration

No parameter configuration is required.

2.15.5.3.7. Dynamic column splitting

Dynamic column splitting allows you to split data records with a dynamic number of columns.

Example

Input data:

```
k1=v1,k2=v2,k3=v3,k4=v4
```

In the preceding example, the data is stored in key-value pairs in the format of key=value. Different data records may have different numbers of key-value pairs, that is, they may have different numbers of columns. In this case, you can use the first-level delimiter, which is comma (,) in the preceding example, to split the data to different key-value pairs. Then, you can use the secondary-level delimiter, which is equal sign (=) in the preceding example, to split each key-value pair to the key and value.

Parameter configuration

Parameter	Description
Select Field	The name of the field to split.
first level column delimiter	The delimiter used to split the field at the first level. Default value: \u0001.
secondary level column delimiter	The delimiter used to split the field at the secondary level. Default value: \u0002.
Add column	The fields that store the split data. Specify a key for each field. An alias is allowed.

2.15.5.3.8. Static column splitting

Static column splitting allows you to split data records with fixed columns that are separated by a fixed delimiter.

Example

You can use commas (,) as the delimiter to split the following data to four new columns, that is, 1111, 2222, 3333, and 4444.

```
1111,2222,3333,4444
```

The static column splitting method is applicable to data records with fixed columns that are separated by a fixed delimiter.

Parameter configuration

Parameter	Description
Select Field	The name of the field to split.
column separator	The delimiter used to split the field. You can use full-width characters or half-width characters as needed.
Add column	The fields that store the split data. Specify a key and a sequence number for each field. An alias is allowed.

2.15.5.3.9. Row splitting

Row splitting allows you to split a row to multiple rows based on a field by using the specified delimiter.

Example

The following table lists the input data.

id	num
1	1,2

Split the row to multiple rows based on the num field by using the comma (,) as the delimiter, and place the split data in the new field new_num. The following table lists the output data.

id	num	new_num
1	1,2	1
1	1,2	2

Parameter configuration

Parameter	Description	Remarks
Select Field	The name of the field to split.	This parameter is set to num in the preceding example.
Field separator	The delimiter used to split the field. Default value: (\n).	This parameter is set to comma (,) in the preceding example.
Define new column name	The name of the new field that stores the split data.	This parameter is set to new_num in the preceding example.

2.15.5.4. Result tables

2.15.5.4.1. Datahub

Datahub is a real-time data distribution platform that is designed to process streaming data. It provides a channel for the Apsara Stack DTplus platform to process big data. Datahub works with multiple Apsara Stack services to provide an end-to-end data processing solution. Realtime Compute typically uses Datahub to store source and result tables for streaming data processing.

Parameter configuration

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.
Output Field	The fields to be returned.
endPoint	The endpoint used to access DataHub. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
project	The name of the Datahub project to which data is to be written. It corresponds to the project parameter in the WITH clause of the CREATE TABLE statement.
topic	The name of the Datahub topic to which data is to be written. It corresponds to the topic parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID used to access Datahub. It corresponds to the accessId parameter in the WITH clause of the CREATE TABLE statement.

Parameter	Description
<code>accessKey</code>	The AccessKey secret used to access Datahub. It corresponds to the <code>accessKey</code> parameter in the WITH clause of the CREATE TABLE statement.
<code>maxRetryTimes</code>	The maximum number of retries for writing data to DataHub. It corresponds to the <code>maxRetryTimes</code> parameter in the WITH clause of the CREATE TABLE statement.
<code>batchSize</code>	The number of data records that are written at a time. It corresponds to the <code>batchSize</code> parameter in the WITH clause of the CREATE TABLE statement.
<code>batchWriteTimeoutMs</code>	The interval at which the cache is cleared. It corresponds to the <code>batchWriteTimeoutMs</code> parameter in the WITH clause of the CREATE TABLE statement.
<code>maxBlockMessages</code>	The maximum number of data blocks that are written at a time. It corresponds to the <code>maxBlockMessages</code> parameter in the WITH clause of the CREATE TABLE statement.

Field type mapping

The following table lists the mapping between Datahub and Realtime Compute data types. We recommend that you declare the type mapping in the DDL statement.

Datahub data type	Realtime Compute data type
BIGINT	BIGINT
DOUBLE	DOUBLE
TIMESTAMP	BIGINT
BOOLEAN	BOOLEAN
DECIMAL	DECIMAL

2.15.5.4.2. Log Service

As an all-in-one real-time data logging service, Log Service allows you to quickly finish tasks such as data ingestion, consumption, delivery, query, and analysis without any extra development work. This can help you improve O&M and operational efficiency, and build up the capability to process large amounts of logs in the data technology era.

Parameter configuration

Parameter	Description
<code>oriTableName</code>	The table name used in the CREATE TABLE statement. It must be a globally unique name.
Output Field	The fields to be returned.

Parameter	Description
endPoint	The endpoint used to access Log Service. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
project	The name of the Log Service project to which the data is to be written. It corresponds to the project parameter in the WITH clause of the CREATE TABLE statement.
primaryKey	The primary key field of the output fields.
source	The name of the log source. It corresponds to the source parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID used to access Log Service.
accessKey	The AccessKey secret used to access Log Service.
mode	The mode of data writing. It corresponds to the mode parameter in the WITH clause of the CREATE TABLE statement. Default value: <code>random</code> . If you set this parameter to <code>partition</code> , data is written by partition.
logStore	The name of the Logstore in the Log Service project to which the data is to be written.

2.15.5.4.3. ApsaraDB RDS

ApsaraDB RDS is a stable, reliable, and scalable cloud database service.

Parameter configuration

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.
Output Field	The fields that you want to write to the table in your ApsaraDB RDS database.
url	The URL used to access your ApsaraDB RDS instance. This parameter corresponds to the url parameter in the WITH clause of the CREATE TABLE statement.
tableName	The name of the table to which you want to write data. This parameter corresponds to the tableName parameter in the WITH clause of the CREATE TABLE statement.
userName	The username that is used to access your ApsaraDB RDS database. This parameter corresponds to the userName parameter in the WITH clause of the CREATE TABLE statement.
password	The password that is used to access your ApsaraDB RDS database. This parameter corresponds to the password parameter in the WITH clause of the CREATE TABLE statement.
maxRetryTimes	The maximum number of retries for writing data to the table. This parameter corresponds to the maxRetryTimes parameter in the WITH clause of the CREATE TABLE statement.

Parameter	Description
batchSize	The number of data records that can be written at a time. This parameter corresponds to the batchSize parameter in the WITH clause of the CREATE TABLE statement.
bufferSize	The maximum number of data records that can be stored in the buffer before deduplication is triggered. This parameter corresponds to the bufferSize parameter in the WITH clause of the CREATE TABLE statement. You can use this parameter only after the primaryKey parameter is specified.
flushIntervalMs	The time interval at which the buffer is cleared. Unit: milliseconds. This parameter corresponds to the flushIntervalMs parameter in the WITH clause of the CREATE TABLE statement.
excludeUpdateColumns	The fields that are not updated when Realtime Compute updates data records with the same primary key value. This parameter corresponds to the excludeUpdateColumns parameter in the WITH clause of the CREATE TABLE statement.
ignoreDelete	Specifies whether to skip delete operations. This parameter corresponds to the ignoreDelete parameter in the WITH clause of the CREATE TABLE statement.
partitionBy	Specifies the partitioning rule for the result table. Before Realtime Compute writes data to the sink node, Realtime Compute performs hash partitioning based on the value of this parameter. The data then flows to the relevant hash node. This parameter corresponds to the partitionBy parameter in the WITH clause of the CREATE TABLE statement.
primaryKey	The primary key field in the output fields.

Data type mapping

ApsaraDB RDS data type	Realtime Compute data type
TEXT	VARCHAR
BYTE	VARCHAR
INTEGER	INT
LONG	BIGINT
DOUBLE	DOUBLE
DATE	VARCHAR
DATETIME	VARCHAR

ApsaraDB RDS data type	Realtime Compute data type
TIMESTAMP	VARCHAR
TIME	VARCHAR
YEAR	VARCHAR
FLOAT	FLOAT
DECIMAL	DECIMAL
CHAR	VARCHAR

JDBC parameters

Parameter	Description	Default value	Required JDBC version
useUnicode	Specifies whether to use the Unicode character set. This parameter must be set to true if you set the characterEncoding parameter to gb2312 or gbk.	<i>false</i>	1.1g
characterEncoding	The character encoding format. This parameter must be set if the useUnicode parameter is set to true. You can set this parameter to gb2312 or gbk.	<i>false</i>	1.1g
autoReconnect	Specifies whether to automatically re-establish a connection if the connection to the database is unexpectedly interrupted.	<i>false</i>	1.1
autoReconnectForPools	Specifies whether to apply the reconnection policy to a database connection pool.	<i>false</i>	3.1.3
failOverReadOnly	Specifies whether to set the connection to read-only after the database is automatically reconnected.	<i>true</i>	3.0.12

Parameter	Description	Default value	Required JDBC version
maxReconnects	The maximum number of reconnection attempts allowed. This parameter must be set if the autoReconnect parameter is set to true.	3	1.1
initialTimeout	The interval between two reconnection attempts. Unit: seconds. This parameter must be set if the autoReconnect parameter is set to true.	2	1.1
connectTimeout	The timeout period when you use a socket connection to access the database server. Unit: milliseconds. Default value: 0. This value indicates that the connection never times out. This parameter applies to JDK 1.4 and later.	0	3.0.1
socketTimeout	The timeout period for read and write operations on a socket connection. Unit: milliseconds. Default value: 0. This value indicates that read or write operations never time out.	0	3.0.1

FAQ

- Q:** When the output data of Realtime Compute is written to an ApsaraDB RDS table, is the result table updated based on the primary key or is a new data record generated in the table?

A: The processing method depends on whether the primary key is defined in the DDL statement.

 - If a primary key is defined in the DDL statement, the result table is updated by using `insert into on duplicate key update`. For a data record, if the primary key does not exist, the record is inserted into the table as a new row. If the value of the primary key field exists, the original row in the table is updated.
 - If no primary key is defined in the DDL statement, new data records are inserted into the table by using `insert into`.
- Q:** What do I need to pay attention to when I perform GROUP BY operations based on the unique index of an ApsaraDB RDS table?

A: An ApsaraDB RDS table has only one auto-increment primary key. Therefore, this auto-increment primary key cannot be declared as the primary key in a Realtime Compute job. If you want to perform GROUP BY operations based on the unique index of the table, declare the unique index as the primary key in the job.

2.15.5.4.4. Tablestore

Tablestore is a NoSQL database service that is built on the Apsara distributed operating system. Tablestore adopts data sharding and load balancing technologies to scale out and handle concurrent transactions. You can use Tablestore to store and access large volumes of structured data in real time.

Parameter configuration

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.
Output Field	The fields that you want to write to the Tablestore table.
instanceName	The name of the Tablestore instance. This parameter corresponds to the instanceName parameter in the WITH clause of the CREATE TABLE statement.
tableName	The name of the table to which you want to write data. This parameter corresponds to the tableName parameter in the WITH clause of the CREATE TABLE statement.
endPoint	The endpoint of the Tablestore instance. This parameter corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID of the account that is used to access the Tablestore instance. This parameter corresponds to the accessId parameter in the WITH clause of the CREATE TABLE statement.

Parameter	Description
accessKey	The AccessKey secret of the account that is used to access the Tablestore instance. This parameter corresponds to the accessKey parameter in the WITH clause of the CREATE TABLE statement.
valueColumns	The names of the fields that you want to insert into the result table. Separate multiple names with commas (,).
bufferSize	The maximum number of data records that can be stored in the buffer before deduplication is triggered. This parameter corresponds to the bufferSize parameter in the WITH clause of the CREATE TABLE statement.
batchWriteTimeoutMs	The write timeout period. This parameter corresponds to the batchWriteTimeoutMs parameter in the WITH clause of the CREATE TABLE statement.
maxRetryTimes	The maximum number of retries for writing data to the table. This parameter corresponds to the maxRetryTimes parameter in the WITH clause of the CREATE TABLE statement.
retryIntervalMs	The retry interval. This parameter corresponds to the retryIntervalMs parameter in the WITH clause of the CREATE TABLE statement.
ignoreDelete	Specifies whether to skip delete operations. This parameter corresponds to the ignoreDelete parameter in the WITH clause of the CREATE TABLE statement.
primaryKey	The primary key field in the output fields.

Data type mapping

Tablestore data type	Realtime Compute data type
INTEGER	BIGINT
STRING	VARCHAR
BOOLEAN	BOOLEAN
DOUBLE	DOUBLE

2.15.5.4.5. MaxCompute

Realtime Compute supports creating a MaxCompute table as the result table.

Parameter configuration

Parameter	Description
-----------	-------------

Parameter	Description
oriTableName	The table name used in the CREATE TABLE statement. It must be a globally unique name.
tableName	The name of the MaxCompute table to which data is to be written.
Output Field	The fields to be written to the MaxCompute table.
endPoint	The endpoint used to access MaxCompute. It corresponds to the endPoint parameter in the WITH clause of the CREATE TABLE statement.
tunnelEndPoint	The endpoint of the Tunnel service, which is required for a MaxCompute project deployed in a Virtual Private Cloud (VPC). It corresponds to the tunnelEndPoint parameter in the WITH clause of the CREATE TABLE statement.
project	The name of the MaxCompute project to which data is to be written. It corresponds to the project parameter in the WITH clause of the CREATE TABLE statement.
accessId	The AccessKey ID used to access MaxCompute. It corresponds to the accessId parameter in the WITH clause of the CREATE TABLE statement.
accessKey	The AccessKey secret used to access MaxCompute. It corresponds to the accessKey parameter in the WITH clause of the CREATE TABLE statement.
partition	<p>The partitions to which the data is to be written. It corresponds to the partition parameter in the WITH clause of the CREATE TABLE statement.</p> <p>This parameter must be specified for a partitioned table. For example, if the partition name of a table is <code>ds=20180905</code>, you can specify the parameter as <code>`partition` = 'ds=20180905'</code>. Separate multiple levels of partitions with commas (.), for example, <code>`partition` = 'ds=20180912,dt=xxxyyy'</code>.</p>

 **Note** Realtime Compute writes cached data to a MaxCompute table every time when a checkpoint is reached.

Field type mapping

MaxCompute data type	Realtime Compute data type
TINYINT	TINYINT
SMALLINT	SMALLINT
INT	INT

MaxCompute data type	Realtime Compute data type
BIGINT	BIGINT
FLOAT	FLOAT
DOUBLE	DOUBLE
BOOLEAN	BOOLEAN
DATETIME	TIMESTAMP
TIMESTAMP	TIMESTAMP
VARCHAR	VARCHAR
STRING	STRING
DECIMAL	DECIMAL
BINARY	VARBINARY

FAQ

Q: Does a real-time computing node clear the result table before it writes data to the MaxCompute sink that is in Stream mode when `isOverwrite` is set to `true` ?

A: The `isOverwrite` parameter is set to `true` by default. That is, a real-time computing node clears the result table and result data before it writes data to the sink. Every time a real-time computing node starts or resumes after being paused, it clears data of the existing result table or the result partition before it writes data. Certain data may be lost when data is cleared after a paused real-time computing node is resumed.

2.15.5.5. FAQ

This topic describes the frequently asked questions (FAQs) about Stream Studio.

Q: What computing engine do I need to activate before using Stream Studio?

A: You must first activate Realtime Compute because Stream Studio is a development platform based on Realtime Compute.

Q: Where can I create a Realtime Compute project? How do I bind the project to Stream Studio?

A: You can create a Realtime Compute project in the Realtime Compute console. After a project is created, you can bind it to an existing DataWorks workspace in the DataWorks console or directly create a workspace and bind the project to it. After the Realtime Compute project is bound to your workspace, you can develop real-time computing nodes in Stream Studio.

Q: What are the advantages of the directed acyclic graph (DAG) mode in Stream Studio? What are the similarities and differences between the DAG mode and SQL mode?

A: Stream Studio supports both the DAG mode and the SQL mode to develop real-time computing nodes. In DAG mode, you can perform drag-and-drop operations on components to configure real-time computing nodes without writing code. In this mode, what you see is what you get. You can also switch to the SQL mode to configure nodes by writing SQL statements.

Q: What types of SQL does Stream Studio support?

A: Realtime Compute is based on Apache Flink. Therefore, Stream Studio supports Flink SQL.

2.16. Data Protection

2.16.1. Overview

Data Protection is a data security management platform. It can be used to detect data assets, detect sensitive data, classify data, de-identify data, monitor data access behavior, report alerts, and audit risks.

Data Protection provides security management services for MaxCompute.

Access Data Protection

1. Log on to the DataWorks console.
2. On the DataWorks page that appears, click the icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now** to go to the Data Protection page.

Features

Data Protection provides the following features:

- **Intelligent sensitive data detection**
Data Protection automatically detects an enterprise's sensitive data based on self-training models and algorithms, and clearly displays statistics on data types, volume, and visitors. It also recognizes custom data types.
- **Accurate data classification:** Data Protection allows you to classify data and create custom levels for better data management.
- **Flexible data de-identification**
Data Protection provides diverse and configurable methods for dynamic data de-identification.
- **Risky behavior monitoring and auditing**
Data Protection uses various correlation analysis algorithms to detect risky behavior. It also provides alerts and supports visualized auditing for detected risks.

2.16.2. Configure rules for defining sensitive data

This topic describes how to configure rules for defining sensitive data.

Procedure

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now** to go to the **Data Security Guard** page.
4. In the left-side navigation pane, choose **Management > Data Definition**. On the Data Recognition Rules page that appears, click **Create Rule**.
5. In the dialog box that appears, set parameters in the **Set Basic Info** step.

You can create a template-based identification rule or a custom identification rule.

Parameter	Description
Data Type	<p>The category of the rule. You can select Template or Custom from the Data Type drop-down list.</p> <ul style="list-style-type: none"> ◦ If you select Template, you can select Personal Information, Merchant Information, or Company Information from the right drop-down list. ◦ If you select Custom, you can enter a data type.
Data Name	<ul style="list-style-type: none"> ◦ If you select Template from the drop-down list, you can select a built-in identification rule template from the right drop-down list. You can select Email, SeatNumber, MobilePhoneNumber, IP, MacAddress, CarNo, PostCode, IdCard, or BankCard. ◦ If you select Custom, you can enter a data name.
Owner	The owner of the rule.
Description	The description of the rule.

6. Click **Next**. Set the **Level** and **Data Definition** parameters.

Parameter	Description
Level	The security level of the sensitive data to which the rule is applied. If the existing security levels cannot meet your needs, click Levels in the left-side navigation pane to change the level settings.

Parameter	Description
Content Scanning	<p>Specifies whether to enable content scanning. This option is selected by default for all the built-in data identification templates.</p> <ul style="list-style-type: none"> ◦ If you select a template, you cannot modify the identification rule, but you can verify the accuracy of the identification rule. ◦ If you select regular expression matching, you can customize the identification rule.
Field Scanning	<p>Specifies whether to enable field scanning. This approach provides two matching methods: exact matching and fuzzy matching of field names. Multiple-field matching is supported, and the relationship between the fields is OR.</p>

7. Click **Next**. After you confirm the configuration, click **Save and Apply**.

-  **Note** When you create a rule to define sensitive data, note the following:
- The rule name must be unique.
 - The content scanning and field scanning configuration must be unique.
 - You can only view the sensitive data that is detected based on the data identification rule one day after the rule takes effect.

2.16.3. View the distribution of sensitive data

On the next day after you configure and activate sensitive data identification rules as a data security administrator, you can access Data Recognition to view the distribution of sensitive data.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. In the left-side navigation pane, click **Data Recognition**. On the Data Recognition page that appears, you can view the overall data distribution and field details.

2.16.4. View the information about data activities

On the next day after you configure and activate sensitive data identification rules as a data security administrator, you can access Data Activities to view related activity statistics, trend, and details.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. In the left-side navigation pane, click **Data Activities** to go to the **Data Activities** page.

The Data Activities page allows you to view the information of each activity that involves sensitive data. On the Manipulations and Queries tab, you can view the statistics, trend, user, and details of data access activities. On the Export tab, you can view the statistics and details of data export.

2.16.5. View the data audited as risky

Data activities are audited manually or based on the risk identification rules and AI-based identification rules. The Data Risks page displays data activities that are audited as risky. You can comment audit results as required.

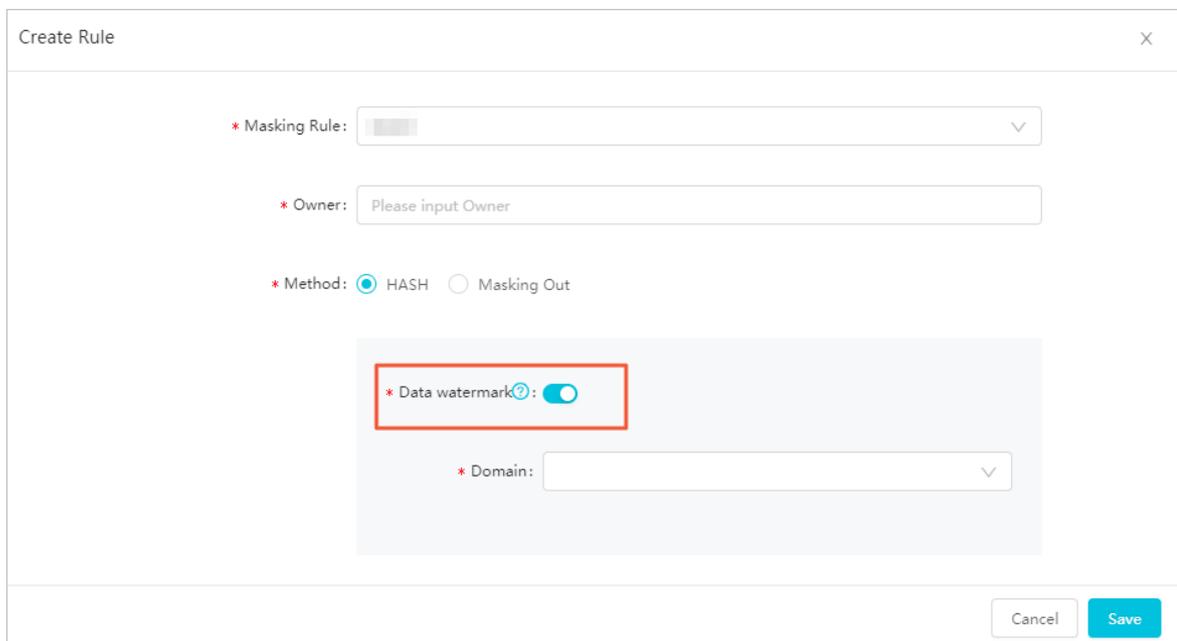
1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now** to go to the **Data Security Guard** page.
4. In the left-side navigation pane, click **Data Risks** to filter and view the data audited as risky as needed.

2.16.6. Track data

You can create a data tracking task to track the source of leaked data. You can also download a data tracking file and delete a data tracking task.

Configure a data watermark

1. [Log on to the DataWorks console](#).
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Data governance > Data Protection**.
3. On the page that appears, click **Try now**.
4. On the Data Protection page, choose **Rule Change > Data Masking** in the left-side navigation pane.
5. On the Data Masking page, click **Create Rule** to create a data masking rule. For more information, see [Customize de-identification rules](#). When you configure a data masking rule, you can turn on **Data watermark**. This way, a data watermark can be embedded into data when data masking is performed on the data, which facilitates the tracking of the leaked data.



 **Note** A data watermark cannot be added to Chinese data.

Create a data tracking task

1. In the left-side navigation pane, click **Data traceability**.

2. On the **Data traceability** page, click **New data tracing task**.
3. Drag the file in which data is leaked to the middle part of the **Traceability tasks** dialog box or click **Upload** to upload the file.

 **Note** You can upload a CSV file whose size is no more than 200 MB.

4. Click **Start tracing** to track the source of the leaked data.

 **Note** After the tracking computing starts, you can close the dialog box, and the computing is not affected.

5. After the computing is complete, click the  icon that corresponds to your data tracking task and view the tracking result. You can also enter a file name in the search box of the **Data traceability** page to view the historical data tracking task.

 **Note** Tracking results are for reference only. A data tracking task may not generate results or generate one or more results.

Download a data tracking file

On the **Data traceability** page, find a data tracking file that you want to download in the **Traceability file** column and click the  icon to download the file.

Delete a data tracking task

1. On the **Data traceability** page, find a data tracking task that you want to delete and click the  icon.
2. In the message that appears, click **Confirm**.

2.16.7. Manage a self-generated data recognition model

This topic describes how to train a model that can be used to summarize the data characteristics of specific columns in a table based on these columns. It also describes how to use the trained model to identify sensitive data.

Go to the Self Generated Data Recognition Model tab

1. [Log on to the DataWorks console](#).
2. On the **DataStudio** page, click the  icon in the upper-left corner and choose **All Products > Data governance > Data Protection**.
3. On the page that appears, click **Try now**.
4. On the **Data Protection** page, choose **Rule Change > Data Recognition Rules** in the left-side navigation pane.
5. Then, click **Self Generated Data Recognition Model**.

Create a self-generated data recognition model

1. On the **Self Generated Data Recognition Model** tab, click **Add Model**.
2. In the **Add Model** dialog box, configure the parameters.

Parameter	Description
Model Name	The name of the model that you want to create. The name can contain letters, digits, and underscores (_).
Owner	The name of the owner for the model. The name of the model can contain letters, digits, spaces, and underscores (_).
Selected Samples	You can select one or more columns from an existing MaxCompute table.

3. Click **Next** to go to the **Model Training** step.
4. Select **I accept data umbrella sampling for model training** and click **Start Training**.

 **Note** The column that you select must contain more than 10 rows. Otherwise, the system cannot start to train the model.

After the training starts, you can close the dialog box and view the training progress in the recognition model list.

5. View the status of the model in the recognition model list. After the training is complete, the status of the model becomes **Training Completed**. Click the  icon in the Actions column. Then, the Assess step of the Edit Model dialog box appears.
6. View the recognition results in the Assess step. In this step, a maximum of 10 results are displayed. You can also adjust the results based on your business requirements. If the accuracy of the recognition can meet your requirements, click **Create**. Then, the model is created. If excessive mismatches exist, you can click **Retrain** to retrain the model after you adjust the recognition results.

 **Note** In most cases, you must perform two to three training to optimize the model.

Retrain the self-generated data recognition model

1. Click **Retrain** to retrain the model if the recognition effect presented by the recognition results in the Assess step is not satisfactory.
2. Navigate to the Select Samples step. The system automatically places the recognition results in the Assess step into the Sample Field and Exclude Field sections based on the recognition results. Click **Next** to continue the training.
3. View the recognition results after the training process is complete.
4. Click **Create** if the recognition results are satisfactory to finish creating the model. If the recognition results are still not satisfactory, click **Retrain** to continue retraining the model.

Use the self-generated data recognition model

1. After the model is created, click **Go To Data Recognition Rules Online Page**.
2. In the **Create Rule** dialog box, Configure the parameters.

Parameter	Description
Data Type	You can set this parameter to Custom or Add By Template .

Parameter	Description
Rule Name	The value cannot be changed.
Owner	The value cannot be changed.
Description	The description of the data recognition rule. The description can be a maximum of 120 characters in length and cannot contain special characters.

3. In the **Specify Details** step, specify **Level** and select **Field Scanning**. Select a trained model from the **Data Recognition Rules** drop-down list and click **Next**.
4. In the **Complete** step, click **Save**.

Stop training a self-generated data recognition model

1. Find a model that is being trained and click the  icon in the **Actions** column to stop training the model.
2. Find the model whose training is stopped and click the  icon to continue the training.

Delete a self-generated data recognition model

Find a model that is not being trained and click the **Delete** icon in the **Actions** column to delete the model.

Note

- A model that is being trained cannot be deleted. If you want to delete such a model, you can stop training the model and delete it.
- A model that is in use cannot be deleted. If you want to delete such a model, you can delete the data recognition rules configured for the model and delete the model.

2.16.8. Manage the data security levels

When creating a rule, you can specify a security level for the data to which the rule applies. On the **Levels** page, you can create and delete security levels. You can also modify the priority of each security level and manage rules by security level.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now** to go to the **Data Security Guard** page.
4. In the left-side navigation pane, choose **Management > Levels**.

On the **Levels** page, you can create and delete security levels. You can also modify the priority of each security level and manage rules by security level.

Operation	Description
Create a security level	Click Create Level . Specify the security level name and operator.

Operation	Description
Manage rules by security level	Find the target security level and click the  icon in the Actions column. In the Manage Rules by Level dialog box that appears, you can select a rule and adjust its security level.
Delete a security level	Find the target security level and click the  icon in the Actions column. In the dialog box that appears, click Delete .
Modify the priority of a security level	Find the target security level. Drag and drop the  icon in the Actions column.

2.16.9. Manage data that is incorrectly detected

On the Manual Check page, you can manually correct the sensitive data that is incorrectly detected by rules. For example, you can delete incorrectly detected data, change the type of the detected data, and delete or recover data in batches.

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now** to go to the **Data Security Guard** page.
4. In the left-side navigation pane, choose **Management > Manual Check**.

On the Manual Check page, you can delete incorrectly detected data, change the type of the detected data, and delete or recover data in batches.

- To delete a data record that is incorrectly detected, turn off the switch in the Status column of the data record.

 **Note** You can recover data records that you have deleted.

- To change the type of a data record, click the edit icon next to the name of the target rule and select a rule.

 **Note** You can only select a rule that has been configured in DataWorks.

- To delete or recover multiple data records at the same time, you can select the data records and click **Remove** or **Recover**.

2.16.10. Customize de-identification rules

This topic describes how to customize de-identification rules in Data Security Guard so that DataWorks can dynamically de-identify the results of ad hoc queries.

Prerequisites

Sensitive data detection rules are created and data security levels are specified. For more information, see [Configure rules for defining sensitive data](#) and [Manage the data security levels](#).

Go to the Data Masking page

1. Log on to the DataWorks console.
2. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now**.
4. In the left-side navigation pane, choose **Management > Data Masking**.

The **Data Masking** page has two tabs: **Data Masking** and **Whitelist**.

Customize de-identification rules in Data Security Guard

1. Set the **Masking Scene** parameter to **Global Config (_default_scene_code)** and click **Create Rule** in the upper-right corner.
2. In the **Create Rule** dialog box, set the **Rule**, **Owner**, and **Method** parameters.

 **Note** Data Security Guard provides three methods for de-identifying ID card numbers and email addresses, including **Pseudonymisation**, **Hashing**, and **Masking Out**. For other types of data, Data Security Guard only provides the **Hashing** and **Masking Out** methods.

o Pseudonymisation

This method replaces the text of a data record with an artificial pseudonym of the same data type. If you select this method, specify a security domain. Rules with different security domains generate different pseudonyms for the same data record.

o Hashing

If you select this method, specify a security domain. Rules with different security domains generate different hash values for the same data record.

o Masking Out

This method uses asterisks (*) to mask specified parts of a data record. It is commonly used.

Parameter	Description
Recommended	You can select recommended policies to mask data of common types such as ID card numbers and bank card numbers.
Custom	You can flexibly specify whether to mask the specified number of characters at the first, middle, or last part of a data record.

3. Click **OK**.
4. On the **Data Masking** tab of the **Data Masking** page, set the status of the de-identification rule to **Active** or **Inactive**.
You can click the **Test** icon in the **Actions** column of the rule to test whether it works.
5. Click the **Whitelist** tab. On the **Whitelist** tab, click **Add Account**.
6. In the **Add Account** dialog box, set the **Rule**, **Account**, **Effective From**, and **To** parameters. For more information about user groups, see [Manage user groups](#).

 **Note** If you query data beyond the time range specified for the whitelist, the query results will be de-identified.

7. Click **Save**.

Verify the de-identification effect in DataWorks

After you create and configure de-identification rules, DataWorks dynamically de-identifies the results of queries in your workspace based on the rules.

 **Note** You must first turn on Mask Data in Page Query Results for your workspace in the DataWorks console.

2.16.11. Manage user groups

You can create a user group on the GroupManagement page and reference it in a de-identification whitelist. You can also copy, edit, and delete user groups on the GroupManagement page.

Go to the GroupManagement page

1. Log on to the DataWorks console.
2. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Data Protection**.
3. Click **Try now**.
4. In the left-side navigation pane, choose **Management > GroupManagement**. The GroupManagement page appears.

Create a user group

1. On the GroupManagement page, click **Create Group** in the upper-right corner.
2. In the **Create Group** dialog box that appears, set the parameters described in the following table.

Parameter	Description
Name	Enter the name of the user group.  Note The name of the user group must be unique.
Owner	Enter the owner of the user group.
Source Type	Specify the source of accounts in the user group. Valid values: <ul style="list-style-type: none">◦ Text: If you select this option, click Upload File next to Source File, select a local file to upload, and then click Open.◦ Select Existing Accounts: If you select this option, select the accounts to add next to Add Members and click >.

3. Click **Save**.

Copy a user group

On the GroupManagement page, find the target user group and click  in the Actions column. An identical user group is generated.

Note

- The name of the generated user group contains the -copy suffix. You can click  to change the name.
- You can only copy the content but not the dependencies of a user group.

Edit a user group

To edit an existing user group, follow these steps:

1. On the **GroupManagement** page, find the target user group and click  in the Actions column.
2. In the **Edit Group** dialog box that appears, modify parameters such as **Name**, **Owner**, and **Source Type**.
3. Verify the settings and click **Save**.

Delete a user group

To delete a user group, find the user group on the GroupManagement page and click **Delete** in the Actions column. In the dialog box that appears, click **Delete**.



Note You cannot delete a user group that is referenced in a de-identification whitelist.

If you still want to delete the user group, delete the user group from the corresponding de-identification whitelist first.

2.17. App Studio

2.17.1. Overview

App Studio is a tool designed to help you develop data products. It comes with a rich set of front-end components that you can drag and drop to simply and quickly build front-end apps.

With App Studio, you do not need to download and install a local integrated development environment (IDE) or configure and maintain environment variables. Instead, you can use a browser to write, run, and debug apps and enjoy the same programming experience as that in a local IDE. App Studio also allows you to publish apps online.

Advantages

App Studio has the following core advantages:

- **Data development anytime, anywhere**
You do not need to download and install a local IDE or configure and maintain environment variables. Instead, you can use a browser to develop data in your office, at home, or anywhere that you can connect to the network.
- **Editor with complete features**
App Studio provides a browser-based editor that allows you to easily write, run, and debug projects. When you enter the code, App Studio provides code hinting, code completion, and repair suggestions. You can also find all references and the definition of a method to automatically generate code.
- **Online debugging**
App Studio comes with all breakpoint types and operations of a local IDE. It supports thread switching and filtering, variable checking and watching, remote debugging, and hot code replacement.

- Multi-feature terminal

You can directly access the runtime environment, which is currently built based on CentOS as the base image. The multi-feature terminal supports all bash commands, including vim and other interactive commands.

- Collaborative coding

You and your team members can use App Studio to share the development environment for collaborative coding. Currently, App Studio allows a maximum of eight users to edit the same file of a project online concurrently, improving work efficiency. In the future, the collaborative coding component will support chatting, bullet screen messages, code annotations, videos, and other features to make teamwork efficient and pleasant.

- Plug-in system

App Studio supports business plug-ins, tool plug-ins, and language plug-ins.

- App Studio allows you to customize any required menu or add any service portal based on your business needs.
- You can customize project management processes, project types, and templates dedicated to your business.
- You can develop common tools, such as enhanced Git features, code rule scanning, keyboard shortcuts, enhanced editing features, and code snippets, and integrate them into App Studio.
- You can use language plug-ins to enrich the languages supported by App Studio, enabling App Studio to serve users with more languages while addressing your own business needs.

- Visual building

App Studio provides a WYSIWYG designer that has rich components and deeply integrates DataService Studio and DataStudio. Among all components of DataWorks, you can call DataWorks API operations only in App Studio. In addition to calling the API operations, you can quickly build front-end apps by dragging and dropping components and configuring them in the WYSIWYG designer based on the santa file system, developing web apps without code.

- Rich templates and flexible project management

App Studio provides rich project templates, allowing you to develop your project accordingly with fewer steps and higher efficiency. You can also save your project as a template for future development and use, or share it with other users.

2.17.2. Get started with App Studio

To build a data portal, engineers need to develop data, build backend services, and develop front-end pages. This topic describes the basic features of App Studio and how to use App Studio.

Originally, DataWorks is mainly used by data engineers to implement offline or streaming data development. As DataWorks becomes increasingly easy to use, many roles such as algorithm engineers, BI analysts, operators, and product managers who are familiar with SQL can use DataWorks to develop data.

App Studio helps different types of users quickly build webpages for data viewing and apps for data query.

Go to the App Studio page

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > App Studio**. The **Projects** page appears.

Create a front-end project

App Studio provides complete front-end development capabilities that allow you to develop front-end projects in the same way as in a local integrated development environment (IDE). Without the need to master or understand any new concepts, you can create front-end projects in App Studio and develop HTML, CSS, JavaScript, and React files in a way that you are familiar with.

1. Create a project based on the sample project.
 - i. Go to the **App Studio** page and click **Projects** in the left-side navigation pane. On the **Projects** page, click **Create Project from Code**.
 - ii. On the **Create Project** page, set the **Name** and **Description** parameters, and set the runtime environment to **react-demo**.

 **Note**

- The name of the project must start with a letter and can contain digits, letters, underscores (_), and hyphens (-).
- The description of the project can be 2 to 500 characters in length.

- iii. After the configuration is completed, click **Submit**.

2. Set running parameters.

In the upper-right corner, choose **Edit Config > Edit Configurations**. In the **Run/Debug Configurations** dialog box that appears, set the required parameter. Select the instance type and specify the port number as required. You can use the default configuration unless otherwise required. Then, click **OK**.

Parameter	Description
Install Cmd	The command used to install the dependency, for example, npm install .
Start Cmd	The command used to start the app, for example, npm start .
Environment Variables	The environment variables.
Initialize Script	The path of the script used to initialize a container in the code library.
PORT	The port of the Elastic Compute Service (ECS) instance. Default value: 3000.
ECS Instance	The instance type. Valid values: 1vCPU 2GMemory , 2vCPU 3GMemory , 4vCPU 8GMemory , and 8vCPU 16GMemory .

3. Run the project.

Click the **Run** icon in the upper-right corner to run the project. Currently, you can run the `tnpm start` command to start front-end projects. You can seamlessly run projects with `webpack-dev-server` configured.

During project running, you can view the dependency installation and app startup logs. After the project running is completed, the **Preview** tab appears in the right-side navigation pane. You can edit and save the code in real time. The edited code takes effect immediately.

4. Access the project.

Click the **Preview** tab in the right-side navigation pane, and click the arrow next to the access link to open the project.

In App Studio, you can edit and develop front-end projects in the same way as in a local IDE. App Studio supports code completion, method signature, refactoring, and redirection for HTML, CSS, LESS, SCSS, JavaScript, TypeScript, JSX, and TSX files. In addition, you can develop front-end projects based on templates without the need to build any environment or download any dependency.

Create a backend project

1. Create a project based on the sample project.
 - i. Go to the **App Studio** page and click **Projects** in the left-side navigation pane. On the **Projects** page, click **Create Project from Code**.
 - ii. On the **Create Project** page, set the **Name** and **Description** parameters, and set the runtime environment to **springboot**.
 - The name of the project must start with a letter and can contain digits, letters, underscores (_), and hyphens (-).
 - The description of the project can be 2 to 500 characters in length.
 - iii. After the configuration is completed, click **Submit**.

2. Set running parameters.

In the upper-right corner, choose **Edit Config** > **Edit Configurations**. In the **Run/Debug Configurations** dialog box that appears, set the required parameter and then click **OK**.

Parameter	Description
Main class	Select the main method. If no main method is available, check whether your project has a main method.
VM options	The virtual machine (VM) options.
Program arguments	The app parameters.
Environment Variables	The environment variables.
JRE	The Java runtime environment (JRE). By default, this parameter cannot be modified.
PORT	The port of the ECS instance. Default value: <i>7001</i> .
ECS Instance	The instance type. Valid values: 1vCPU 2GMemory , 2vCPU 3GMemory , 4vCPU 8GMemory , and 8vCPU 16GMemory .
Pre-Launch Option	The commands to be run before the project is run. You can specify up to three commands.
Enable Hot Code	Specifies whether to enable hot code replacement.

You can click **Add** on the left of the **Run/Debug Configurations** dialog box to add multiple configurations for running.

3. Run the project.

Click the **Run** icon in the upper-right corner to run the project.

The first time that the project is run takes a longer time because App Studio needs to allocate the ECS instance and initialize the language service. After the running is completed, the Runtime tab appears, showing the access link.

4. Access the project.

Click **Open Link** to access the project.



Append /testapi to the link and refresh the page.



Understand App Studio

The following operations are supported for created projects:

- Top navigation bar

- Project

From the Project menu, you can configure the project or view detailed information by selecting **Character Set** or **Project Information**. Provided information about the current project includes the ID specified by **Project ID**, name specified by **Project Name**, type specified by **Project Type**, creation time specified by **Created At**, and **UUID**.

- File

From the File menu, you can create a file or open a recently created file by selecting **Create File** or **Re-Open Most Recent Files**.

- Edit

From the Edit menu, you can perform common editing operations. To search all the code in the project and open the related file, select **Find in Path**.

- **Version**

From the Version menu, you can select **Switch Branch**, **View Changes**, **Submit**, **View Log**, **Connect to Remote Repo**, and **Merge Abort**.

- **Switch Branch**

In the Check Out Branch dialog box, you can click **+ Create Branch** to create a local branch and push it to the remote repo. You can click a local branch and select **checkout** from the shortcut menu on the right to switch to the branch. You can also select **merge** to merge the selected branch to the current branch.

You can click a remote branch and select **check out as a new local branch** from the shortcut menu on the right to check out the remote branch locally. Then rename the branch. You can also select **merge** to merge the selected branch to the current branch.

- **View Changes**

Click **View Changes** to view the list of edited files on a local branch in the right-side navigation pane.

- **Submit**

Click **Submit** to commit edits on a local branch for staging. You must enter the commit information.

- **View Log**

On the Log page, you can view all commit records of branches and filter them.

- **Connect to Remote Repo**

You can associate a new project with a remote repo for version control.

- **View**

You can click **Toggle Full Screen** or press **Esc** on the keyboard to enter or exit the full screen mode of the page. You can also click **Hide Sidebar** or **Hide Status Bar** to hide the right-side navigation pane or the status bar. If they are hidden, you can click **Show Sidebar** or **Show Status Bar** to show them respectively.

- **Debug**

- If you create a front-end project, you can set running parameters and add custom images.
 - App Studio supports Java-based debugging. In addition to setting running parameters and adding custom images, you can perform many other operations for debugging backend projects. You can also perform full or incremental builds and compile the `Main.java` file.

- **Settings**

From the Settings menu, you can set the Git configuration to import the Git code to create a project. You can also configure your preference and shortcut keys.

- **Deploy**

You can choose **Deploy > Download Source Code** to download the source code.

- **Template**

You can choose **Template > Manage Templates** to go to the **My Templates** page to manage templates.

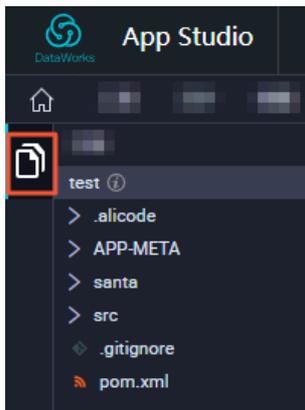
- **Left-side navigation pane**

- **Entry**

Click the icon framed in red. The project section appears.

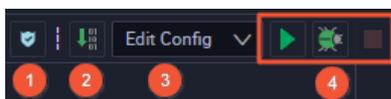
- **Edit section**

Double-click a file that you want to edit. In the Edit section that appears, right-click the code section to perform the following operations.



Action	Description
Go to Definition	Navigates to the definition page.
Peek Definition	Previews the definition.
Find All References	Searches for all references.
Workspace Symbol	Searches for a symbol in the project.
Go to Symbol...	Navigates to the symbol in the project.
Generate...	Generates the code.
Rename Symbol	Renames the symbol.
Change All Occurrences	Changes the name of all occurrences of a symbol throughout the file.
Format Document	Formats the file.
Cut	Cuts the file.
Copy	Copies the file.
Command Palette	Goes to the command palette.

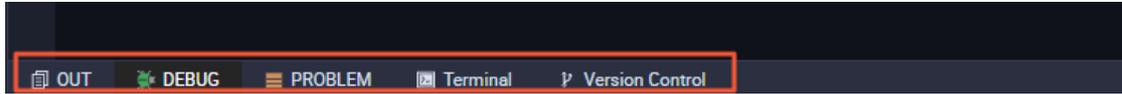
- Icons in the upper-right corner



No.	Feature
1	Alibaba Coding Guidelines
2	Build Program. You can perform this operation only when the project is running or being debugged.
3	Run/Debug Configurations. You can set parameters for running or debugging the project.

No.	Feature
4	Operations on the project, including running, debugging, or stopping the project.

- Bottom bar

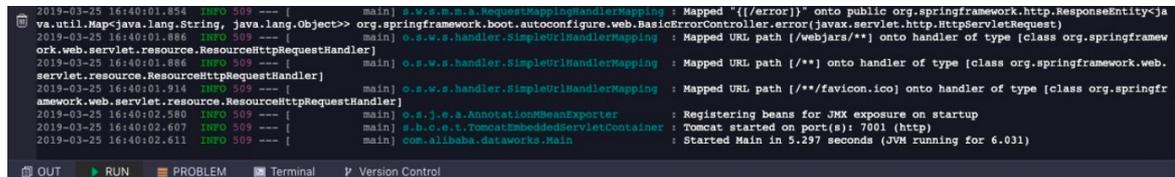


- OUT tab

You can click the OUT tab to view the output.

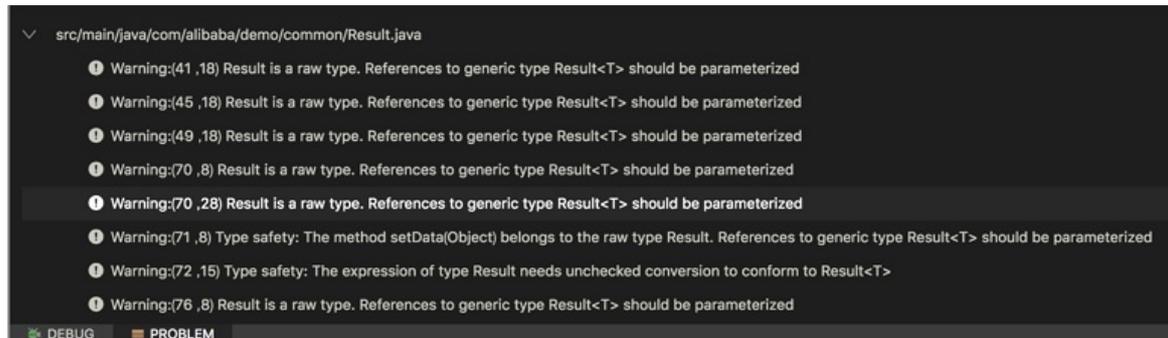
- RUN or DEBUG tab

If you click the Run or Debug icon for a project, this tab appears, showing the progress and information of the project.



- PROBLEM tab

If you click the Run or Debug icon for a project that has a problem, this tab appears.



- Terminal tab

When running or debugging a project, you can click the Terminal tab and run bash or vim commands on the ECS instance.



- Version Control tab

You can click the Version Control tab to view the logs and history of the project.

2.17.3. Navigation pane

2.17.3.1. View and manage projects

You can create and manage projects on the Projects page.

Go to the App Studio page and click **Projects** in the left-side navigation pane. On the page that appears, you can view projects that you have created. For more information about how to create template-based and code-based projects, see [Project management](#).

Click a project to go to the project editing page. You can also click **Create Template** of a project to create a template based on the project.

Create a template

1. Click **Create Template** of a project.
2. In the **Create Template** dialog box that appears, set each parameter.

Parameter	Description
Name	The name of the template.
Description	The description of the template.
Class	The class of the template.

3. After the configuration is completed, click **OK**.

2.17.3.2. View and manage templates

You can view all templates created based on projects on the Templates page.

Click a template to go to the template details page. Then, click **Code Editor** to view the project code that this template is based on.

You can also click **Create Project** of a template to create a project based on this template.

2.17.4. Project management

This topic describes how to create and manage projects.

You can create a template-based or code-based project.

Create a template-based project

1. Go to the App Studio page and click **Projects** in the left-side navigation pane. On the **Projects** page, click **Create Project from Template**.
2. On the **Create Project** page, specify **Name** and **Description**, and select a template.

Note

- You can select a custom template or a template provided by the system.
- All projects created by using templates support WYSIWYG development.

3. After the configuration is completed, click **Submit**.

Create a code-based project

You can create a project by running code. App Studio provides code templates for three types of runtime environments. Select a code template as required.

1. Go to the App Studio page and click **Projects** in the left-side navigation pane. On the **Projects** page, click **Create Project from Code**.
2. On the **Create Project** page, specify **Name** and **Description**, and select a template.
3. After the configuration is completed, click **Submit**.

View and manage projects

You can view the created projects on the **Projects** page.

You can click a project name to go to the project editing page. You can also click **Create Template** of a project to create a template based on the project.

 **Note** You can view projects shared by others but cannot create templates based on those projects.

2.17.5. Code editing

2.17.5.1. Overview

Code editing supports common IDE features, such as automatic completion, code hinting, syntax diagnosis, and global content search.

The following tables list the basic and advanced features that App Studio supports in different languages.

Basic feature	Java	Python	JavaScript and TypeScript
Completion	Supported	Supported	Supported
Hover	Supported	Supported	Supported
Diagnostics	Supported	Supported	Supported
SignatureHelp	Supported	Supported	Supported
Definition	Supported	Supported	Supported
References	Supported	Supported	Supported
Implementation	Supported (coming soon)	Not supported	Not supported
DocumentHighlight	Supported	Supported	Supported
DocumentSymbol	Supported	Supported	Supported
WorkspaceSymbol	Supported	Supported	Supported
CodeAction	Supported (Alibaba Java Guidelines coming soon)	Supported	Supported
CodeLens	References implementation	Not supported	Not supported
Formatting	Supported	Supported	Not supported

Basic feature	Java	Python	JavaScript and TypeScript
RangeFormatting	Supported	Not supported	Not supported
FindInPath	Supported	Supported	Supported

Advanced feature	Java	Python	JavaScript and TypeScript
Rename	Supported	Supported	Supported
WorkspaceEdit	Supported	Not supported	Not supported
UnitTest (quick start)	Supported	Not supported	Not supported
MainClass	Supported	Not supported	Not supported
MainClassQuickStart	Not supported	Not supported	Not supported
ListModules	Supported	Not supported	Not supported
Generate	Constructor Override Getter and Setter Implement	Not supported	Not supported

2.17.5.2. Generate code snippets

Currently, App Studio supports the Java class constructor, getter and setter methods, override methods of the parent class that a child class inherits, and API methods to be implemented.

Entry

Perform either of the following operations to generate the Java code:

- Right-click the code section and select **Generate**.
- Press Command+M on the keyboard. The Java code is automatically generated.

Constructor

On the Generate menu, click **Constructor**.

Select the fields to be included in the constructor and click **OK**.

The constructor that contains the initialization statement of the fields is generated.

Getter and setter methods

Generate the getter and setter methods in a way similar to the constructor.

 **Note** If a Java class does not have any field or the Java class is overwritten by the @data annotation of lombok, the getter or setter method is not required for the Java class. In this case, the Getter, Setter, and Getter And Setter options do not appear on the Generate menu.

Override methods

Click **Override Methods** on the **Generate** menu. All methods that can be overridden are listed in the **Generate Code** dialog box.

Select a method. The corresponding method is generated.

2.17.5.3. Run UT

App Studio currently supports unit testing (UT), including automatically generating UT code, detecting the entry for UT, running UT code, and displaying the UT result.

Automatically generate UT code

Open the target file, right-click the code editing section, select **Generate** and then click **Create Test**. The UT class file and UT code are automatically generated in the test directory.

Detect the entry for UT

 **Note**

- UT class files must be stored in the `src/test/java` directory. A Java UT class file that is not stored in this directory cannot be identified as the Java UT class.
- For a method annotated with `@Test` annotation, **Run Test** appears, indicating the entry for UT.

After the Java UT class file is created, add the `@Test` annotation of `org.junit.Test` to the corresponding sample UT method.

Run UT code

Click the **Run** icon in the upper-right corner. The sample UT starts.

2.17.5.4. Find in Path

App Studio provides the **Find in Path** feature to support global content search.

Move the pointer over **Edit** in the top navigation bar and select **Find in Path**.

You can select **Match Case**, **Words**, **Regex**, and **File Mask** as required. If you select **File Mask**, you must also select a file name extension from the right drop-down list to search in files of the specified type.

You can also search for content in the specified project, module, or directory.

After selecting a file, you can locate the searched content in the file and open the file in the editor.

2.17.6. Debugging

2.17.6.1. Configuration and startup

You can configure the entry method, start debugging, and set breakpoints to debug an app.

Configure the entry method

Parameter	Description
Main class	The entry method (which is the main method) you want to start. You can select a value from the drop-down list.

Parameter	Description
VM options	The parameters for starting a Java Virtual Machine (JVM), for example, -D, -Xms, and -Xmx.
Program arguments	The startup parameter, which is obtained by the args parameter in the main method.
Environment Variables	The environment variables.
JRE	The Java runtime environment. Default value: <i>1.8 - SDK</i> .
PORT	The port you want to expose in the app, for example, classic port 7001 or port 8080 for Spring Boot-based projects.
ECS Instance	The type of the ECS instance used for debugging.
Enable Hot Code	This configuration takes effect only in Run mode. By default, the HotCode2 plug-in that Alibaba Cloud provides is used.

Start debugging

Move the pointer over **Debug** in the top navigation bar and click **Start Debugging**.

The first startup is slower, because the system needs to prepare the runtime environment and download Maven dependencies for you. When you restart debugging, App Studio skips this process and provides user experience similar to that in a local IDE.

2.17.6.2. Online debugging

App Studio supports the online debugging of Java apps and Spring Boot-based web projects.

Before online debugging, you must configure the entry method and start debugging. For more information, see [Configuration and startup](#).

Exposed services

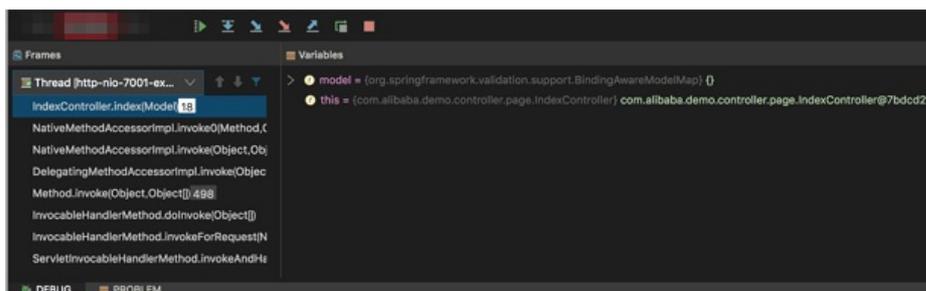
After your app is started, two basic services are provided. You can click the link next to Backend to debug the back-end Java code.

Panel introduction

- Output

The Output panel displays the standard output, excluding System.in, of all apps. It supports the ANSI color and guarantees consistent experience as a local terminal.

- Call Stack

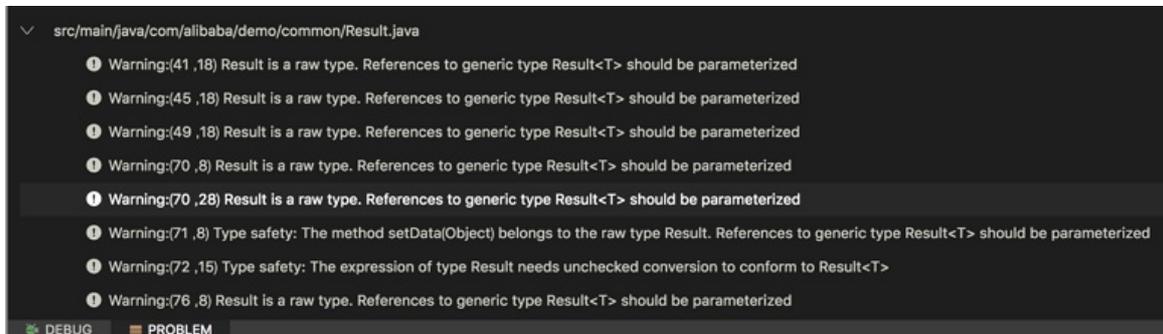


- Breakpoint

The Breakpoint panel displays the breakpoints that are currently set. For more information about the breakpoint types and usage, see Breakpoint types.

- **PROBLEM**

The **PROBLEM** panel displays compilation problems of apps. You can click a record to go to the corresponding line in the file.



2.17.6.3. Breakpoint types

App Studio supports normal line breakpoints, method breakpoints, and exception breakpoints.

Normal line breakpoint

You can click the blank section next to a line in the current file to generate a breakpoint for that line. The breakpoint also appears on the Breakpoint panel.

Method breakpoint

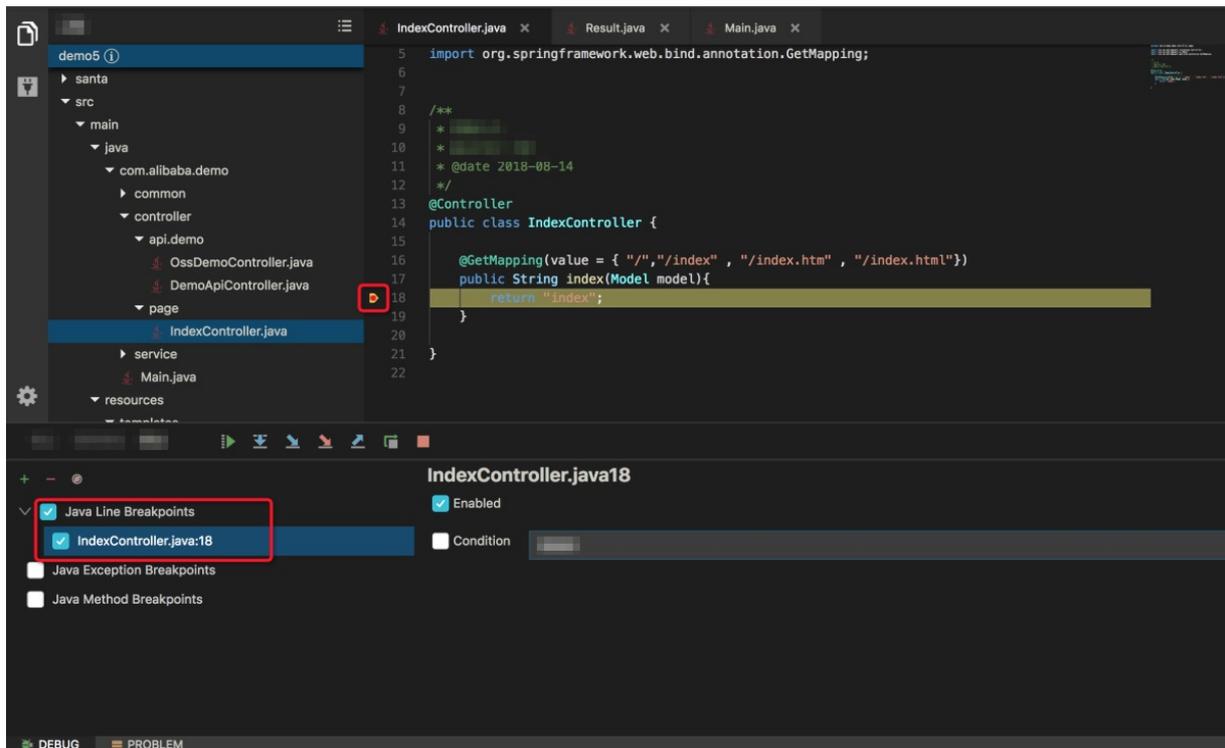
Different from a line breakpoint or an exception breakpoint, a method breakpoint triggers two events, namely, entry and exit. You can manually add a method breakpoint, or set a breakpoint at the place where the method is defined.

If the method breakpoint is triggered, the program stops when stepping into or out of the method.

Exception breakpoint

If an exception breakpoint is set, the program stops when encountering the exception.

As shown in the following figure, after index is triggered, the program stops in line 23 because **NullPointerException** appears.



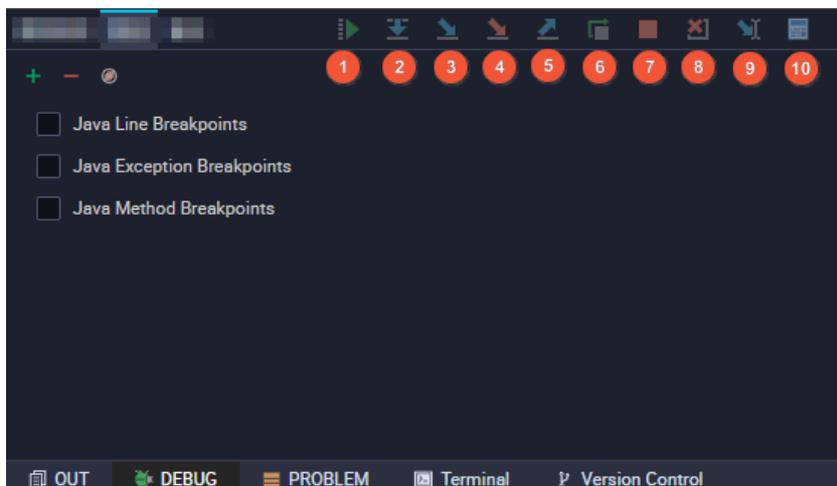
2.17.6.4. Breakpoint operations

The Breakpoint panel displays the breakpoints that are currently set. This topic describes how to operate breakpoints.

Breakpoints can be classified into normal line breakpoints, method breakpoints, and exception breakpoints. For more information, see [Breakpoint types](#).

Debugging buttons

You can perform the debugging operations by clicking the following buttons listed in the table:



No.	Feature	Description
1	Continue	Resumes the current breakpoint to continue the current thread.
2	Step Over	Runs to the next line.
3	Step Into	Steps into a method.
4	Force Step Into	Forcibly steps into a method of a class not to be stepped into. Different from Step Into , Force Step Into enables you to step into a method from a built-in Java library.
5	Step Out	Steps out of the current method.
6	Restart	Currently, the Restart button is not perfect enough and may not be able to clean up the program. This button is being optimized.
7	Stop	Stops debugging.
8	Drop Frame	Deletes the current stack and returns to the previous method.
9	Run to Cursor	Runs to the current line of code. You can set a temporary breakpoint in a line.
10	Evaluate Expression	Calculates an expression.

2.17.6.5. Terminal

You can start multiple terminals in App Studio.

The **Terminal** tab appears in the lower part of the page.

App Studio supports common shell commands such as **ls** and **cat** and interactive commands such as **vi** and **top**.

2.17.6.6. Hot code replacement

Using the hot code replacement feature, you can edit the running code of an app and make the edits effective without restarting the app.

For example, after you edit the code while debugging a Spring Boot-based app, you do not need to restart the app. The edited code takes effect once it is saved. App Studio supports this feature by default.

App Studio also supports hot code replacement while an app is running. To trigger hot code replacement, you only need to save the file without installing any plug-in or manually compiling the file.

If you are editing the code in Debug mode, App Studio automatically deletes the current running stack and returns to the method entry.

Configure hot code replacement in Run mode

Enable hot code replacement on the Run/Debug Configurations page.

After you click Run or Debug, the output information of the HotCode2 plug-in appears on the OUT tab.

Save the file after editing it.

Configure hot code replacement in Debug mode

You can use the native Java Debug Interface (JDI) to enable hot code replacement in Debug mode. However, due to Java Virtual Machine (JVM) restrictions, hot code replacement is unavailable when a method is added to or deleted from a class. You can save the file to trigger hot code replacement.

 **Note** The native JVM supports hot code replacement for operations such as adding or deleting a class. However, hot code replacement is unavailable when you change the class structure.

2.17.7. WYSIWYG designer

2.17.7.1. Get started with the WYSIWYG designer

This topic describes basic operations in the WYSIWYG designer, including creating a project and building a visual page.

Create a project

1. Log on to the DataWorks console.
2. On the DataStudio page that appears, click the DataWorks icon in the upper-left corner and choose **All Products > App Studio**. The **Projects** page appears.
3. Click **Projects** in the left-side navigation pane. On the page that appears, click **Create Project from Code**.
4. On the **Create Project** page, set the **Name** and **Description** parameters, and set **Select the runtime environment** to **appstudio**.
5. After the configuration is completed, click **Submit**.

Build a visual page

Open a project created by using the WYSIWYG designer. Go to the *santa/pages* directory in your project.

Double-click a .santa file to go to the WYSIWYG designer. For example, you can double-click the file named *home.santa*.

You can also right-click **pages** and choose **Create > Template** to develop the page based on a template.

The WYSIWYG designer consists of the component menu and operation panel.

- Component menu

The component menu lists all components that the WYSIWYG designer presets, including **layout components**, **basic components**, **form components**, **chart components**, and **advanced components**.

Select a component from the component menu and drag and drop it to the visual operation section. Click the component. The **Component Settings** panel appears on the right.

On the **Component Settings** panel, you can configure the component on the **Properties**, **Style**, and **Advance** tabs.

- Operation panel

You can click the corresponding icon on this panel to **undo an operation**, **redo an operation**, **preview the rendering result**, **enable the code mode**, **use the global style**, **configure the navigation**, **configure a global data flow**, **deploy as a template**, and **save edits**.

Click the **Configure Navigation** icon in the upper-right corner to go to the navigation configuration page. For more information, see [Navigation configuration](#).

Configure a global data flow

For more information about how to configure a global data flow, see [Global data flow](#).

On the Component Settings panel, you can configure the component on the **Properties**, **Style**, and **Advance** tabs.

- Configure component properties

On the Properties tab, you can visually configure component properties.

Based on the rules for configuring component properties, a visual form is generated on the Properties tab. After you configure component properties in this form, the WYSIWYG designer re-renders the component in the visual operation section based on the new properties. You can view the rendering results of the component with different properties in real time.

- Configure component styles

On the Style tab, you can configure the styles of a component.

A visual panel for configuring common styles is provided on the Style tab. On this panel, you can customize the basic styles of a component, including the layout, text, background, border, and effect.

After you add or modify the component styles on this tab, the WYSIWYG designer collects all the style settings and re-renders the component in the visual operation section based on the new component style. You can view the component configuration effect in real time.

- Configure association between components

On the Advanced Settings tab, you can configure association between components.

Select a component in the visual operation section and click the **Advance** tab. The properties of the selected component are listed on the left of the tab. Click the Magnifier icon on the right and select the component to be associated to your selected component.

The properties of the associated component appear on the right of the tab.

Select a property, for example, searchParams, in the left property list and connect it to a property, for example, requestParams, in the right property list.

In this way, any change of the searchParams parameter of the left component is transferred to the requestParams parameter of the right component in real time. This achieves property-based association between the two components.

Configure the code mode

By using the code mode, you can implement complex interactions in a more advanced way. For more information, see [Code mode](#).

Save, preview, run, and hot code replacement

For more information, see [Save, preview, run, and hot code replacement](#).

2.17.7.2. Code mode

By using the code mode, you can implement complex interactions in a more advanced way.

Click the **Code Mode** icon in the upper-right corner of the operation panel to enable the code mode.

The WYSIWYG designer uses domain-specific language (DSL) at the intermediate layer to switch between the visualization mode and code mode. DSL can be considered as a simplified version of React. The DSL syntax is basically the same as the React syntax.

As shown in the code section in the preceding figure, DSL uses a tag to describe a component. The tag properties are the component properties. The property value can be of a simple data type such as a string or a number. The property value can also be an expression. You can enter `state.xxx` to obtain data from the global data flow.

The code mode has the following features:

- If you drag and drop a component or configure the component properties in the visualization section, the edits are updated in the code in real time.
- If you edit the code in the code section, the edits are updated in the visualization section in real time.
- The drag-and-drop operation and component property configuration in the visualization section and code edits in the code section can be converted between each other.

2.17.7.3. DSL syntax

Domain-specific language (DSL) is a component-based language developed based on the features of React JSX and Vue templates and is more suitable for UI layout design.

JSX

The DSL syntax is similar to the JSX syntax in the `React.render` method. The following section provides a brief description of JSX:

- You can use `{}` to switch an HTML scope to a JavaScript scope. In a JavaScript scope, you can write any valid JavaScript expression. The return value appears on the page, for example, `<div>{'Hello' + ' Relim'}</div>`.

 **Note** You can write any JavaScript expressions such as computing statements or literals in `{}`.

- An HTML tag is used to switch a JavaScript scope to an HTML scope, for example, `<div>Hello Relim</div>`.
- The HTML scope and JavaScript scope can be nested, for example, `<div>{'Hello' + ' Relim'}</div>`.

Valid JavaScript expressions

```
// Computing statements
{aaa} // Variable aaa must be defined.
{aaa * 111}
{1 == 1 ? 1 : 0}
{/^123/.test(aa)}
{{1,2,3}.join("")}
{(()=>{return 1})()} // The self-executing function.
// Literals
{1}
{true}
{{11,22,33}}
{{aa:"11",bb:"22"}}
{()=>1} // Describe a function, which is valid but meaningless.
```

Note If certain complex logic must be implemented by multiple computing statements rather than only one statement, you can wrap the logic in a self-executing function, which must be a valid expression. The following statements provide an example:

```
{{function(){
  // Sum the even digits of a number array.
  var input = [1,2,3,4,5,6,7,8,9,10];
  var temp = input.filter(i => i % 2 == 0)
  return temp.reduce((buf, cur) => buf + cur, 0)
}}()
```

Invalid JavaScript expressions

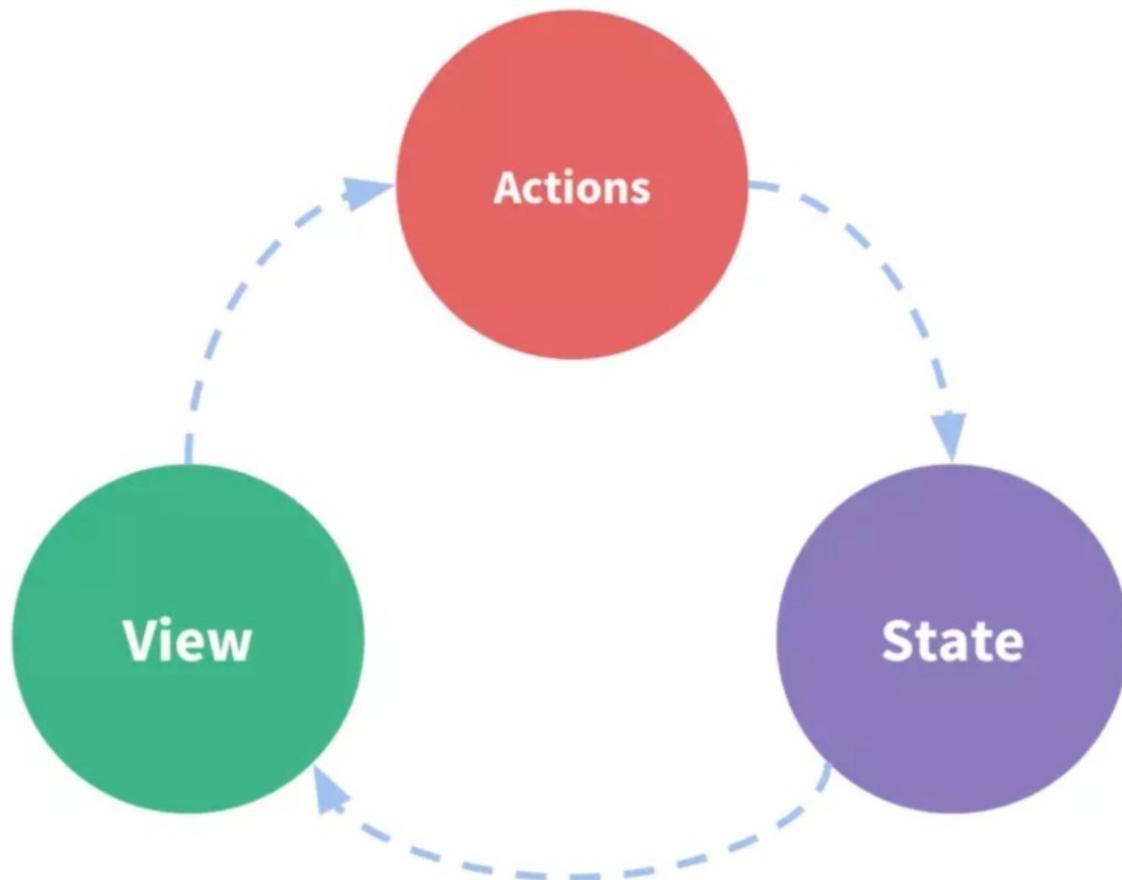
```
{ var a = 1 } // The value assignment statement.
{ aaa * 111; 2 } // Multiple statements separated with semicolons (;).
```

2.17.7.4. Global data flow

A global data flow is used for front-end data management. For multiple components that need to share a state, it is difficult to transfer the state among them. To resolve this issue, you can extract the shared state and use a global data flow to transfer it to all related components.

Principles

In a global data flow, global data is transferred in a globally unique way. Once the data declared in global data changes, the data flow shown in the following figure is executed.



1. A component triggers an action when, for example, a user clicks the component.
2. The action triggers global data changes.
3. Upon the global data changes, components that reference the global state are automatically re-rendered.

Scenarios

A global data flow is applicable to the association of two or more components on a page. You can refine public data into global data for unified management, and then use a global data flow to associate two or more components.

Configure a global data flow

1. Click the **Global Data Flow Settings** icon in the upper-right corner of the operation panel.
2. In the **Global Data Flow Settings** dialog box that appears, set **Variable Name** and **Value**.
 - The variable value can be a number, character string, or JSON string.
 - If the variable value is declared as an API endpoint, data obtained from the API is automatically used as the value of the variable name.
3. Click **Save**.

Use a global data flow

- Obtain global data

Use `state.name` in the component to obtain global data.

```
<Input value={state.name} />
```

- Modify global data

Use the `$setState()` method in the component to modify global data.

```
<Input onChange={value => $setState({ name: value })} />
```

 **Note** You must use the `$setState()` method to modify global data. If you use `state.name = 'new value'`, re-rendering cannot be triggered.

2.17.7.5. Save, preview, run, and hot code replacement

In the WYSIWYG designer, you can perform operations such as saving edits, previewing the rendering result, running an app, or making edits in hot code replacement mode.

Save edits

The WYSIWYG designer periodically saves your edits. You can also click the **Save** icon in the upper-right corner of the operation panel to save edits.

Preview the rendering results

In the WYSIWYG designer, code in the operation section is in the editable status. However, special processing is added for the editable status of some components. For these components, you can run the rendering logic only when the app is running. To preview the rendering result, click the **Preview** icon in the upper-right corner of the operation panel.

Run an app

In the WYSIWYG designer, you can open and edit only one santa file at a time. To view the effect of the entire app,

click the **Run Program** icon on the **Debug** panel of **App Studio** to run the app.

Make edits in hot code replacement mode

If you are not satisfied with any page after running the app, you can edit the code in the WYSIWYG designer and save the edits.

The edited code takes effect on the running page in hot code replacement mode.

2.17.7.6. Navigation configuration

This topic describes how to configure the site navigation in the WYSIWYG designer.

The WYSIWYG designer provides each app with a public page header, a public bottom bar, and public sidebars, where you can configure various menus and themes. You can also specify whether to display the public header, bottom bar, and sidebars as required.

Click the **Navigation Settings** icon in the upper-right corner of the operation panel to go to the page for configuring the navigation of an app.

Configure the public header

You can configure the public header based on your business requirements.

Parameter	Description
Enabled	Specifies whether to display the public header.
Theme	The theme of the public header. You can select a dark or light theme.
Logo Image	The logo image of the site. You can enter an image URL or upload a local image.
Title	The title of the site.
Fix to Page Top	Specifies whether to fix the public header to the top of the page. If you turn on this switch, the public header stays at the top of the page when the page scrolls.
Menu Items	The menu items such as the link name and link URL that are displayed in the public header.

Configure the sidebars

You can configure the sidebars based on your business requirements.

Parameter	Description
Enabled	Specifies whether to display the sidebars.
Theme	The theme of the sidebars. You can select a dark or light theme.
Enable Folding	Specifies whether the sidebar menus can be hidden.

2.18. Migration Assistant

2.18.1. Overview

The Migration Assistant service of DataWorks allows you to migrate data objects across different DataWorks versions, Alibaba Cloud accounts, regions, and workspaces.

Migration Assistant allows you to export data objects in your workspace, including auto triggered nodes, manually triggered nodes, resources, functions, data sources, table metadata, ad hoc queries, and components. You can create full export tasks, incremental export tasks, or custom export tasks to export your data objects in DataWorks based on your business requirements.

 **Notice** To create export or import tasks, you must use an Alibaba Cloud account or be the workspace administrator. If you use a Resource Access Management (RAM) user that is not assigned the administrator role, you can only view export and import tasks.

Scenarios

- Back up node code

You can use Migration Assistant to periodically back up your node code to prevent data from being deleted by mistake. In this case, we recommend that you create a full export task.

- Export a common workflow for replication

You can use Migration Assistant to export a common workflow that can be replicated in other workspaces. In this case, we recommend that you create a custom export task.

- **Build a test environment**

You can use Migration Assistant to copy all the node code and replace the production data with test data to build a test environment. In this case, we recommend that you create a full export task or a custom export task.

- **Develop data in a hybrid cloud environment**

You can use Migration Assistant to migrate node code from Alibaba Cloud public cloud to Alibaba Cloud Apsara Stack to develop data in a hybrid cloud environment. We recommend that you create a custom export task. If the difference between Alibaba Cloud public cloud and Alibaba Cloud Apsara Stack is large, a compatibility problem may occur when data objects are migrated.

- **Migrate data objects between the development environment and production environment**

If a workspace consists of the production environment and development environment that are completely isolated, you can use Migration Assistant to export nodes from the development environment and import them to the production environment for deployment.

2.18.2. Cloud tasks

2.18.2.1. Export tasks from open source engines

DataWorks allows you to migrate tasks from open source scheduling engines such as Oozie and Azkaban to DataWorks. This topic describes the requirements for the files to be exported.

Context

Before you import a task of an open source scheduling engine to DataWorks, you must export the task to your on-premises machine or Object Storage Service (OSS). For more information about the import procedure, see [Import tasks of open source engines](#).

Export a task from Oozie

Requirements and structure of the package to be exported:

- **Requirements**

The package must contain XML-formatted definition files and configuration files of a flow task. The package is exported in the ZIP format.

- **Structure**

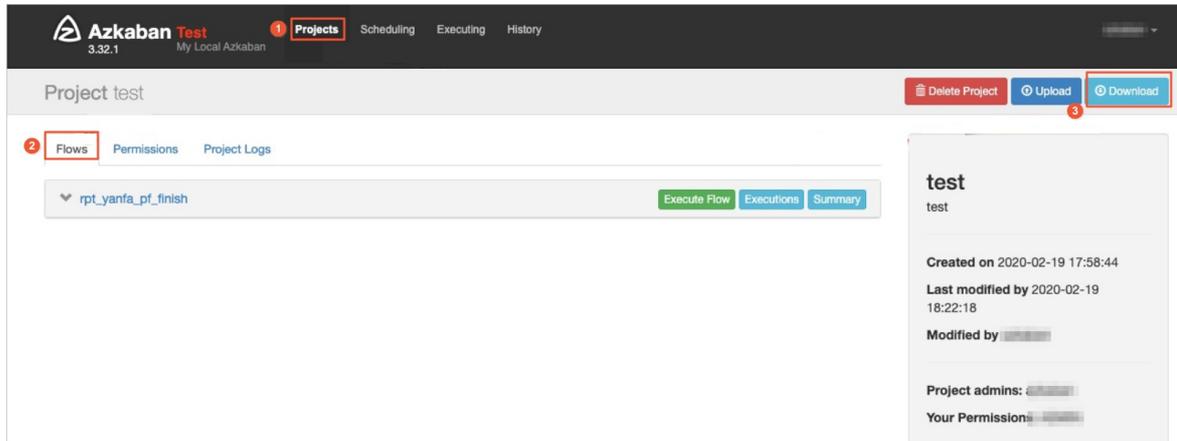
Oozie task descriptions are saved in a Hadoop Distributed File System (HDFS) directory. For example, each subdirectory under the apps directory in the Examples package at the Apache Oozie official website is a flow task of Oozie. Each subdirectory contains XML-formatted definition files and configuration files of a flow task.

```
1 $tree
2 .
3 |— aggregator
4 |   |— coordinator-with-offset.xml
5 |   |— coordinator.xml
6 |   |— job.properties
7 |   |— job-with-offset.properties
8 |   |— lib
9 |   |   |— oozie-examples-4.2.0.jar
10 |   |— workflow.xml
11 |— sqoop
12 |   |— db.hsqldb.properties
13 |   |— db.hsqldb.script
14 |   |— job.properties
15 |   |— workflow.xml
16 |— cron
17 |   |— coordinator.xml
18 |   |— job.properties
19 |   |— workflow.xml
20 |— cron-schedule
21 |   |— coordinator.xml
22 |   |— job.properties
23 |   |— workflow.xml
```

Export a task from Azkaban

You can download a specific flow task in the Azkaban console.

1. Log on to the Azkaban console and go to the **Projects** page.
2. Select a project whose package you want to download. On the page for the project, click **Flows** to show all flow tasks under the project.
3. Click **Download** in the upper-right corner of the page to download the package of the project.



Native Azkaban packages can be exported. No limit is imposed on the packages of Azkaban. The exported package in the ZIP format contains information about all tasks and relationships under a specific project of Azkaban.

Export tasks from other open source engines

DataWorks provides a standard template for you to export the tasks of open source engines except for Oozie and Azkaban. Before you run an export task, you must download the standard template and modify the content to be exported based on the file structure in the template. You can go to the **Open Source engine export** page to download the standard template and view the file structure.

1. [Log on to the DataWorks console](#).
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Other > Migration Assistant**.
3. In the left-side navigation pane, choose **Cloud tasks > Open Source engine export** to go to the **Open Source engine export scheme selection** page.
4. Click the **Standard Template** tab.
5. On the **Standard Template** tab, click **standard format Template** to download the template.
6. Modify the content to be exported based on the template and generate a package to be exported.

2.18.2.2. Import tasks of open source engines

This topic describes how to import tasks that are exported from open source engines into DataWorks.

Procedure

1. Go to the **Open Source engine import** page.
 - i. [Log on to the DataWorks console](#).
 - ii. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Other > Migration Assistant**.
 - iii. In the left-side navigation pane, choose **Cloud tasks > Open Source engine import**.
2. Create an import task.

- i. On the **Import Tasks** page, click **Create Import Task** in the upper-right corner.
- ii. In the **Create Import Task** dialog box, configure the parameters as required.

Parameter	Description
Name	The name of the import task. The name can contain only letters, digits, underscores (_), and periods (.).
Engine type	The engine type. Valid values: Azkaban , Oozie , and Standard format .
Upload From	<p>The source of the package that you want to import. Valid values: Local and OSS.</p> <ul style="list-style-type: none"> ▪ If you select Local for this parameter, perform the following steps to upload a package on your machine: <ol style="list-style-type: none"> a. Click Upload File. b. Select the package that you want to upload and click Open. c. Click Check. d. After the message The resource package has passed the check appears, verify that the file format and content are correct. <div style="background-color: #e1f5fe; padding: 5px; margin: 5px 0;"> <p> Note The size of the package that you want to upload cannot exceed 30 MB. If the size of the package exceeds 30 MB, select OSS for this parameter.</p> </div> <ul style="list-style-type: none"> ▪ If you select OSS for this parameter, enter the endpoint of an Object Storage Service (OSS) object in the OSS Endpoint field. Then, click Check and Preview in sequence to check and preview the package that you want to upload.
Remarks	The description of the import task.

iii. Click **OK**. The **Edit import task** page appears.

3. Edit the import task.

- i. On the **Edit import task** page, specify **Import objects**. **Periodic tasks** is selected for **Import objects** by default. If you want to import data objects of another type, select the required value from the **Import objects** drop-down list.
- ii. (Optional) Click **Advanced Settings**. In the **Advanced Settings** dialog box, configure the mappings between the compute engine instances and the node types and click **OK**. If multiple compute engine instances are bound to the destination workspace, you must complete the settings in the **Advanced Settings** dialog box. You can configure the mappings between compute engine instances and nodes of the **Shell**, **Hive**, and **Sqoop** types.
- iii. On the **Edit import task** page, click **start import** in the upper-right corner.

4. View the import report.

- i. In the **Import progress** dialog box, confirm the import task progress.
- ii. After the import task is completed, click **Return to import task list**.
- iii. Find the task on the **Import Tasks** page and click **View Import Report** in the **Actions** column. On the page that appears, view the task information in the **Basic Information**, **Import Settings**, **Import results**, and **Details** sections.

2.18.3. Migrate data objects in DataWorks

2.18.3.1. Create and view export tasks

Migration Assistant allows you to export data objects in your workspace, including auto triggered nodes, manually triggered nodes, resources, functions, table metadata, data sources, components, and ad hoc queries. This topic describes how to create and view export tasks.

Prerequisites

To create export or import tasks, you must use an Alibaba Cloud account or be the workspace administrator. If you use a Resource Access Management (RAM) user that is not assigned the administrator role, you can only view export and import tasks.

Context

Migration Assistant allows you to export data objects in different modes. These modes include full export, incremental export, and custom export. You can choose an export mode that best suits your business scenario.

- Full export tasks are used to export all the data objects in a workspace. For example, you can run a full export task to back up node code or clone the workspace to a test environment. When you run a full export task, data objects of the latest version are exported.

Only saved data objects can be exported. If a node is saved in both the development environment and production environment, the node saved in the development environment is exported.

- Incremental export tasks are used to export data objects that were modified after the specified date.

 **Note** You cannot configure a blacklist for incremental export tasks.

- Custom export tasks are used to export data objects that you specify. For example, you can run a custom export task to extract a common workflow and clone it to other workspaces. If a workspace runs in both a production environment and a development environment that are completely isolated from each other, you can run a custom export task to export nodes from the development environment and import them to the production environment for deployment.

Go to the Migration Assistant page

1. [Log on to the DataWorks console.](#)
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Other > Migration Assistant**. The **DataWorks export** page under **DataWorks migration** appears.

Create a full export task

1. On the **Export Tasks** page, click **Create Export Task** in the upper-right corner.
2. In the **Create Export Task** dialog box, configure the parameters as required.

Parameter	Description
Name	The name of the export task. The name can contain only letters, digits, underscores (_), and periods (.).
Type	The type of the export task. Select Full export for this parameter. A full export task is used to export all the auto triggered nodes, manually triggered nodes, table metadata, and data sources that have been saved or committed in the current workspace.

Parameter	Description
Blacklist	Specifies whether to enable the blacklist feature for full export based on your business needs. If you select the Add to Blacklist check box, you can add the nodes and resources that do not need to be exported to a blacklist.
Export Version	Valid values: Standard , Private Cloud(>=V3.12) , and Private Cloud(V3.6.1-V3.1.1) . The DataWorks version determines the format in which data objects are exported. Check the DataWorks version of the destination workspace before you create an export task.
Remarks	The description of the export task.

3. (Optional)Click **Add to Blacklist** and run the export task. If you select **Add to Blacklist**, perform the following steps to configure the blacklist:
 - i. In the **Create Export Task** dialog box, click **Add to Blacklist**.
 - ii. On the **Set Blacklist** page, select the data objects that you do not want to export.
 - iii. Click **Add Selected to Blacklist**.
 - iv. Click **Export** in the upper-right corner.
 - v. In the **Export confirmation** message, click **Confirm**.
4. (Optional)If you do not select **Add to Blacklist**, click **Export** in the **Create Export Task** dialog box.
5. In the **Export Progress** dialog box, view the progress of the export task. After the task succeeds, click **Back to Export Tasks**.

Create an incremental export task

1. On the **Export Tasks** page, click **Create Export Task** in the upper-right corner.
2. In the **Create Export Task** dialog box, configure the parameters as required.

Parameter	Description
Name	The name of the export task. The name can contain only letters, digits, underscores (_), and periods (.).
Type	The type of the export task. Select Incremental for this parameter. An incremental export task is used to export data objects that were modified after the specified date. Supported data objects include auto triggered nodes, manually triggered nodes, table metadata, and data sources that have been saved or committed in the current workspace.
Start Date	The date on which data is modified. The data generated after this date is incremental data.
Export Version	Valid values: Standard , Private Cloud(>=V3.12) , and Private Cloud(V3.6.1-V3.1.1) .
Remarks	The description of the export task.

3. Click **Export**.

Create a custom export task

1. On the **Export Tasks** page, click **Create Export Task** in the upper-right corner.
2. In the **Create Export Task** dialog box, configure the parameters as required.

Parameter	Description
Name	The name of the export task. The name can contain only letters, digits, underscores (_), and periods (.).
Type	The type of the export task. Select Custom for this parameter. A custom export task is used to export data objects that you specify. Supported data objects include auto triggered nodes, manually triggered nodes, table metadata, and data sources that have been saved or committed in the current workspace.
Export Version	Valid values: Standard , Private Cloud(>=V3.12) , and Private Cloud(V3.6.1-V3.1.1) .
Remarks	The description of the export task.

3. Click **Select Export Objects**.
4. On the **Export Objects** page, select a type of data object that you want to export from the **Export Object** drop-down list. Valid values of **Export Object**: **Table**, **Periodic tasks**, **Resources**, **Manual tasks**, **Function**, **DATA_SERVICE**, **Data source**, **Components**, and **Temporary query**.
5. Select the data objects that you want to export and click **Add Selected to Export Package**. You can also configure filter conditions such as **Export Object**, **Object Type**, and **Export Environment** to search for data objects. Then, click **Add All to Export Package** to add all the data objects that have been found to the package that you want to export.
6. Click **Export** in the upper-right corner.

View and manage export tasks

On the **Export Tasks** page, you can view the name, type, creator, status, update time, and description of created export tasks. The operations that you can perform on export tasks vary based on their status.

- If an export task is in the **Successful** state, you can perform the following operations on the task:
 - Click **View Export Report** in the **Actions** column. On the page that appears, view the task information in the **Basic Information**, **Overview**, and **Details** sections.
 - Click **Download** in the upper-right corner to download the package of the export task to a local directory.
 - Clone the export task.
 - Full export task for which the blacklist feature is not enabled: Click **Clone** in the **Actions** column. In the **Clone** dialog box, specify **Name** and click **Export**.
 - Full export task for which the blacklist feature is enabled: Click **Clone** in the **Actions** column. In the **Clone** dialog box, specify **Name** and click **Add to Blacklist**.
 On the **Set Blacklist** page, select the data objects that you do not want to export, click **Add Selected to Blacklist**, and then click **Export** in the upper-right corner.
 - Custom export task: Click **Clone** in the **Actions** column. In the **Clone** dialog box, specify **Name** and click **Select Export Objects**.
 On the **Export Objects** page, select the data objects that you want to export, click **Add Selected to Export Package**, and then click **Export** in the upper-right corner.
- If an export task is in the **Export failed** state, you can click **View Export Package**, **Download Export Package**, or **Re-export** in the **Actions** column as required. To retry the export task, click **Re-export**.
- If an export task is a custom export task that is in the **Editing** state, you can perform the following operations on the task:

- Click **Edit** in the Actions column. On the **Export Objects** page, modify the data objects that you want to export.
- Click **View Export Package** in the Actions column. On the **Export Package Details** page, view the task information in the **Basic Information**, **Overview**, and **Details** sections.
- Click **Delete** in the Actions column. In the **Delete** message, click **Ok** to delete the task.
- If an export task is a full export task that is in the **Editing** state, you can click **Edit**, **Delete**, or **View Blacklist** in the Actions column as required. If you click **View Blacklist** in the Actions column, you can check the blacklist and click **Export** to run the export task or click **Close** in the dialog box that appears.

2.18.3.2. Create and view import tasks

After you run an export task to export data objects from a workspace, you can create an import task to import these data objects to a specified workspace.

Prerequisites

To create export or import tasks, you must use an Alibaba Cloud account or be the workspace administrator. If you use a Resource Access Management (RAM) user that is not assigned the administrator role, you can only view export and import tasks.

Go to the Migration Assistant page

1. [Log on to the DataWorks console.](#)
2. On the DataStudio page, click the  icon in the upper-left corner and choose **All Products > Other > Migration Assistant**. The **DataWorks export** page under **DataWorks migration** appears.

Create an import task

1. In the left-side navigation pane of Migration Assistant, choose **DataWorks migration > DataWorks import**.
2. On the **Import Tasks** page, click **Create Import Task** in the upper-right corner.
3. In the **Create Import Task** dialog box, configure the parameters as required.

Parameter	Description
Name	The name of the import task. The name can contain only letters, digits, underscores (_), and periods (.).

Parameter	Description
Upload From	<p>The source of the package that you want to import. Valid values: Local and OSS.</p> <ul style="list-style-type: none"> ◦ If you select Local for this parameter, perform the following steps to upload a package on your machine: <ul style="list-style-type: none"> a. Click Upload File. b. Select the package that you want to upload and click Open. c. Click Check. d. After the message The resource package has passed the check appears, click Preview. On the page that appears, check the package that you want to import. <div style="background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Note The size of the package that you want to upload cannot exceed 30 MB.</p> </div> <ul style="list-style-type: none"> ◦ If you select OSS for this parameter, enter the endpoint of an Object Storage Service (OSS) object in the OSS Endpoint field. Then, click Check and Preview in sequence to check and preview the package that you want to upload.
Remarks	The description of the import task.

4. Click **OK**. The **Import Task Settings** page appears. Make sure that you have checked the format and content of the package before you click **OK**.
5. Configure the import task. When you configure the import task, you must complete the settings in the **Engine Instance Mapping** section. The settings in other sections are optional and can be configured as required.
 - (Optional)
 - i. In the **Engine Instance Mapping** section, select a compute engine of the destination workspace for each compute engine that is bound to the source workspace. If multiple compute engines are bound to the source workspace and only one compute engine is bound to the destination workspace, the import task fails. This is because some types of nodes cannot be created in the destination workspace due to the absence of the required compute engines.
 - ii. (Optional) In the **Resource Group Mapping** section, configure resource group mapping between the source and destination workspaces. This ensures that resource groups are available for running imported nodes.
 - iii. (Optional) In the **Dependency Mapping** section, configure workspace mapping for relevant nodes. Some nodes use the name of the source workspace in their code. In this case, you must configure workspace mapping for the nodes to run properly after they are imported. Set the **New Workspace** parameter to the name of the destination workspace. The system uses this workspace name to replace the original workspace name in the node code and the names of the ancestor and descendant nodes of the current node. After the import task is complete, the original workspace name is replaced with the new workspace name.
 - iv. (Optional) In the **Dry-run** section, find the destination node that you want to set as a dry-run node and click **Set up empty run** in the Actions column. You can also select multiple nodes and click **Batch Configure** to set these nodes as dry-run nodes.

This configuration is used to configure the scheduling mode of auto triggered nodes. The auto triggered node that is set as a dry-run node returns a success response without running and does not generate data.

- v. (Optional) In the **Commission Rules** section, configure the commission rules for **Resources**, **Tables**, and **Functions**, and specify whether to enable **Change Owner** as required.

 **Note**

- If a data object with the same name as the data object you want to import exists in the destination workspace, the imported data object cannot be committed.
- If you select No for the Change Owner parameter and no owner is specified for the node you want to import, you are automatically configured as the owner of the node after it is imported.

6. Click **Import** in the upper-right corner.
7. In the **Confirm** message, click **OK**.

View and manage import tasks

On the **Import Tasks** page, the operations that you can perform on import tasks vary based on their status.

- After an import task is complete, you can view the details of the task. To view the task details, find the task on the **Import Tasks** page and click **View Import Report** in the Actions column. On the page that appears, view the task information in the **Basic Information**, **Import Settings**, **Import results**, and **Details** sections.
- If an import task is in the **Editing** state, you can perform the following operations on the task:
 - Click **Edit** in the Actions column. On the **Import Task Settings** page, modify the task configurations.
 - Click **Preview** in the Actions column. On the page that appears, view the task information in the **Basic Information**, **Overview**, and **Details** sections.
 - Click **Delete** in the Actions column. In the message that appears, click **Ok** to delete the task.
- If an import task is in the **Import failed** state, you can click **Re-import** in the Actions column of the task. In the **Import progress** dialog box, click **Return to import task list** after the import task is complete.

2.19. Workspace management

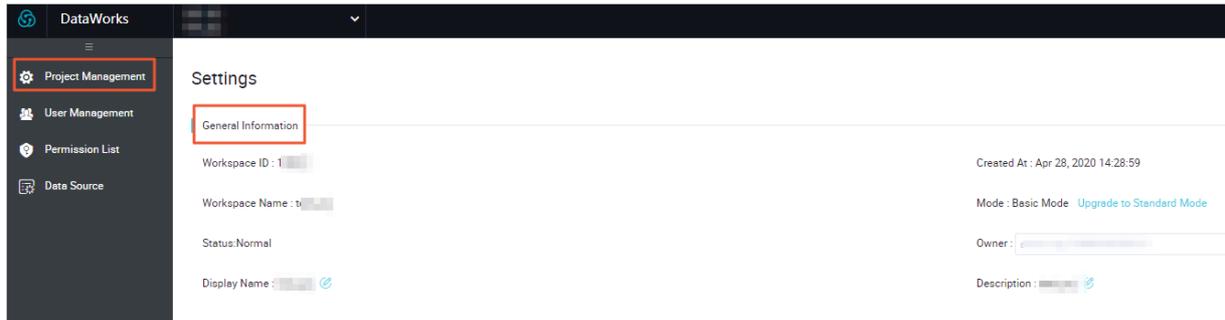
2.19.1. Configure a workspace

On the **Project Management** page of a workspace, you can manage and configure the workspace.

Go to the Project Management page

1. Log on to the DataWorks console.
2. On the **DataStudio** page, click  in the upper-right corner.
3. On the **Project Management** page, configure the workspace as required in the **Basic properties**, **Scheduling properties**, **Security Settings**, and **Compute Engine information** sections.

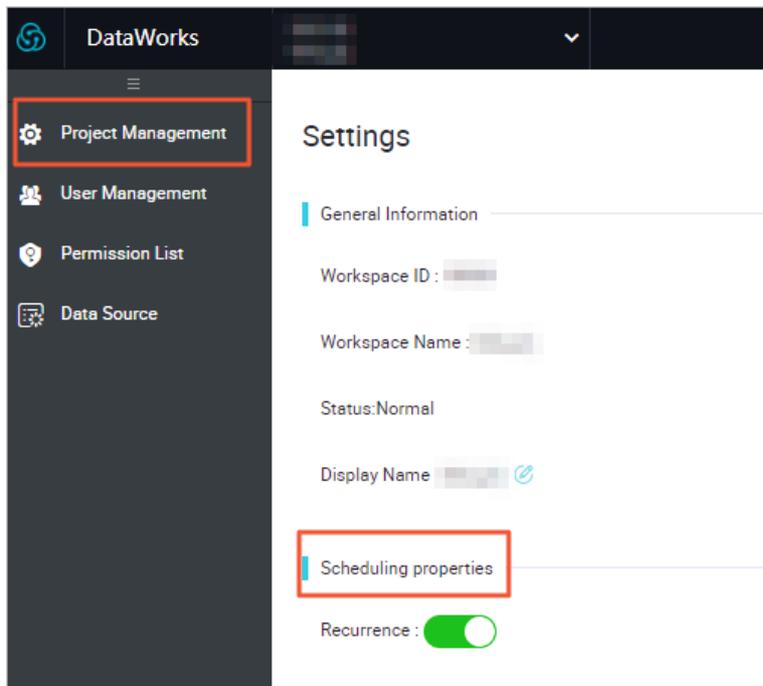
Basic properties section



Parameter	Description
Workspace ID	The ID of the workspace.
Workspace name	The name of the workspace. The name must start with a letter and can contain only letters and digits. It is not case-sensitive. It uniquely identifies the workspace and cannot be changed after the workspace is created.
Status	The status of the workspace.
Display name	The display name that is used to identify the workspace. It can contain only letters and digits. You can change the display name as required.
Creation date	The time when the workspace was created, which cannot be changed.
Mode	The mode of the workspace. Valid values: Basic Mode and Standard Mode .
The head of	The owner of the workspace. The owner of the workspace cannot be changed.
Description	The description of the workspace, which provides comments on the workspace. You can modify the description as required. The description can be up to 128 characters in length and can contain letters, special characters, and digits.

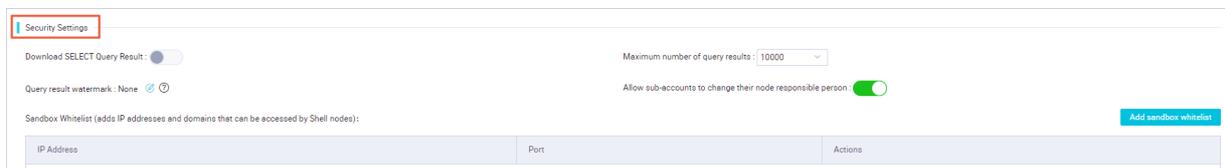
Scheduling properties section

In the **Scheduling properties** section, you can specify whether to enable the recurrence feature for the current workspace.



Recurring nodes can be run in this workspace only after you turn on the **Enable scheduling cycle** switch.

Security Settings section



Parameter	Description
Allow download of select results	Specifies whether to allow workspace members to download data query results in DataStudio. If you turn on this switch, workspace members can download data query results returned by SELECT statements in DataStudio.
Maximum number of query results	The maximum number of data records that can be returned for each query. Valid values: 10, 100, 500, 1000, 5000, and 10000 . Default value: 10000 . Assume that you set the Maximum number of query results parameter to 1000 . Go to the DataStudio page 5 minutes later. In the left-side navigation pane, click Ad-Hoc Query . On the Ad-Hoc Query tab, create an ODPS SQL node and run the node to query data in a table that has more than 1,000 data records. 1,000 records are returned.
Query result watermark	Specifies whether to display watermarks for the query results.
Allow sub-accounts to change their own node owners	Specifies whether to allow RAM users to change the owner for their nodes.

Parameter	Description
Sandbox whitelist (configure IP addresses or domain names that shell tasks can access)	The whitelist of IP addresses and domain names that a Shell node run on the default resource group can access.

To add an IP address to the whitelist, perform the following steps:

1. In the **Security Settings** section, click **Add sandbox whitelist**.
2. In the **Add sandbox whitelist** dialog box, enter the IP address and port number in the **Address** and **Port** fields respectively.
3. Click **Confirm**.

Computing Engine information section

DataWorks supports the following types of compute engines: MaxCompute, E-MapReduce, Realtime Compute, and Graph Compute.

MaxCompute

1. Click the **MaxCompute** tab in the **Computing Engine information** section. On this tab, you can view the information about all available MaxCompute instances in the current workspace.
2. Click **Add instances**.
3. In the **Add MaxCompute engine** dialog box, set the **MaxCompute project name** and **MaxCompute access identity** parameters.
4. Click **Confirm**.

E-MapReduce

1. Click the **E-MapReduce** tab in the **Computing Engine information** section. On this tab, you can view the information about all available E-MapReduce clusters in the current workspace.
2. Click **Add instances**.
3. In the **New EMR cluster** dialog box, set relevant parameters.

Parameter	Description
Instance display name	The display name of the E-MapReduce cluster to be added.
Region	The region where the current workspace resides.
Access ID	The AccessKey ID of the account that is authorized to access the E-MapReduce cluster.
Access Key	The AccessKey secret of the account that is authorized to access the E-MapReduce cluster.
Cluster ID	The ID of the E-MapReduce cluster.
EmrUserID	The ID of the user who created the E-MapReduce cluster.
Project ID	The ID of the project in the E-MapReduce cluster.
YARN resource queue	The name of the resource queue in the E-MapReduce cluster. Unless otherwise specified, set the value to <i>default</i> .

Parameter	Description
Endpoint	The endpoint of the E-MapReduce cluster. You can obtain the endpoint from the E-MapReduce console.

4. Click **Confirm**.

Realtime Compute

1. Click the **Real-time computing** tab in the **Computing Engine information** section. On this tab, you can view the information about all available Realtime Compute projects in the current workspace.
2. Click **Add engine service**.
3. In the **New Blink engine** dialog box, set the **Enter the Realtime Compute project name to add parameter**.

 **Notice** A Realtime Compute project can be bound to only one DataWorks workspace. After a Realtime Compute project is bound to a DataWorks workspace, the Realtime Compute project cannot be used in other DataWorks workspaces. After you bind a Realtime Compute project to a workspace, you cannot unbind it. Use caution when you bind a Realtime Compute project.

4. Click **Binding**.

Graph Compute

1. Click the **GraphCompute** tab in the **Computing Engine information** section.
2. Click **Bind a GraphCompute instance**.

 **Notice** A Graph Compute instance can be bound to only one DataWorks workspace. After a Graph Compute instance is bound to a DataWorks workspace, the instance cannot be used in other DataWorks workspaces.

3. In the **Bind a GraphCompute instance** dialog box, set relevant parameters.

Parameter	Description
Instance display name	The display name of the Graph Compute instance to be bound to the workspace.
GraphCompute instance name	The name of the Graph Compute instance.

4. Click **Binding**.

2.19.2. Manage workspace members

This topic describes how to add and delete members and change the roles of members on the User Management page.

Go to the User Management page

1. Log on to the DataWorks console.
2. On the **DataStudio** page, click  in the upper-right corner.
3. In the left-side navigation pane, click **User Management**. You can view the information about each member in the following columns on the **User Management** page:

- **Members:** the display name of the Apsara Stack tenant account or RAM user that the member uses to log on to DataWorks.
- **Cloud account:** the Apsara Stack tenant account or RAM user that the member uses to log on to DataWorks.
- **Role:** the roles of the member in the current DataWorks workspace. A member can have one or more of the following roles: workspace owner, workspace administrator, developer, administration expert, deployment expert, visitor, and security expert.

 **Note** The workspace administrator can assign roles to RAM users as required. You can click **Permission List** in the left-side navigation pane to view the specific permissions of different member roles.

- **Join TIME:** the time when the member was added to the workspace.
- **Operation:** the operation that you can perform on the member. You can delete all members except the workspace owner from the workspace.

Add members

1. On the **User Management** page, click **Add Member** in the upper-right corner.
2. In the **Add Member** dialog box, select one or more accounts to add in the **Account to be added** list.
3. Click **>** to move the selected accounts to the **Added account** list.
4. Select the roles next to **Batch role setting**. You can select one or more of the following roles: **Space Administrator**, **Development**, **Operation & Maintenance (O & M)**, **Deployment**, **Visitors**, and **Security Administrator**.
5. Click **Confirm**.

Remove a member

1. On the **User Management** page, find the target member and click **Delete** in the **Operation** column.
2. In the **Member remove this workspace** message, click **OK**.

Assign a role to a member

Click the drop-down list in the **Role** column of the member and select the role to be assigned.

Revoke a role from a member

Click **x** next to the role in the **Role** column.

2.19.3. Permission list

DataWorks provides seven roles: workspace owner, workspace administrator, developer, administration expert, deployment expert, visitor, and security expert. You cannot grant the role of the workspace owner to other workspace members. This topic describes the permissions of these roles. In the following tables, Yes indicates that a role has the corresponding permission, and No indicates that a role does not have the corresponding permission.

Data management

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Delete self-created tables	Yes	Yes	Yes	No	No	No	No
Specify categories for self-created tables	Yes	Yes	Yes	No	No	No	No
View favorite tables	Yes	Yes	Yes	No	No	No	No
Create tables	Yes	Yes	Yes	No	No	No	No
Unhide self-created tables	Yes	Yes	Yes	No	No	No	No
Modify the schemas of self-created tables	Yes	Yes	Yes	No	No	No	No
View self-created tables	Yes	Yes	Yes	No	No	No	No
View the content of self-submitted permission requests	Yes	Yes	Yes	No	No	No	No
Hide self-created tables	Yes	Yes	Yes	No	No	No	No
Specify the time-to-live (TTL) for self-created tables	Yes	Yes	Yes	No	No	No	No
Request permissions on tables created by others	Yes	Yes	Yes	No	No	No	No
Delete tables	No	Yes	Yes	No	No	No	No
Update tables	No	Yes	Yes	No	No	No	No
Preview data	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Preview table data of other organizations	Yes	Yes	No	No	No	No	No

Deployment management

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Create deployment tasks	Yes	Yes	Yes	Yes	No	No	No
View the list of deployment tasks	Yes	Yes	Yes	Yes	Yes	Yes	No
Delete deployment tasks	Yes	Yes	Yes	Yes	No	No	No
Run deployment tasks	Yes	Yes	No	Yes	Yes	No	No

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the content of deployment tasks	Yes	Yes	Yes	Yes	Yes	Yes	No

Buttons

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Button: Stop	Yes	Yes	Yes	No	No	No	No
Button: Format	Yes	Yes	Yes	No	No	No	No
Button: Edit	Yes	Yes	Yes	No	No	No	No
Button: Run	Yes	Yes	Yes	No	No	No	No
Button: Zoom In	Yes	Yes	Yes	No	No	No	No
Button: Save	Yes	Yes	Yes	No	No	No	No
Button: Show/Hide	Yes	Yes	Yes	No	No	No	No
Button: Delete	Yes	Yes	Yes	No	No	No	No

Code development

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Save and commit nodes	Yes	Yes	Yes	No	No	No	No
View the code of nodes	Yes	Yes	Yes	Yes	Yes	Yes	No
Create nodes	Yes	Yes	Yes	No	No	No	No
Delete nodes	Yes	Yes	Yes	No	No	No	No
View the node list	Yes	Yes	Yes	Yes	Yes	Yes	No
Run nodes	Yes	Yes	Yes	No	No	No	No
Edit the code of nodes	Yes	Yes	Yes	No	No	No	No
Download files	Yes	Yes	Yes	No	No	No	No

Function development

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View function details	Yes	Yes	Yes	Yes	Yes	Yes	No
Create functions	Yes	Yes	Yes	No	No	No	No
Query functions	Yes	Yes	Yes	Yes	Yes	Yes	No
Delete functions	Yes	Yes	Yes	No	No	No	No

Node types

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Node type: Machine Learning	Yes	Yes	Yes	No	No	No	No
Node type: ODPS MR	Yes	Yes	Yes	Yes	Yes	Yes	No
Node type: Data Sync	Yes	Yes	Yes	Yes	Yes	Yes	No
Node type: ODPS SQL	Yes	Yes	Yes	Yes	Yes	Yes	No
Node type: XLIB	Yes	Yes	Yes	Yes	Yes	Yes	No
Node type: Shell	Yes	Yes	Yes	Yes	Yes	Yes	No
Node type: Zero-Load Node	Yes	Yes	Yes	Yes	Yes	Yes	No

Resource management

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the resource list	Yes	Yes	Yes	Yes	Yes	Yes	No
Delete resources	Yes	Yes	Yes	No	No	No	No
Create resources	Yes	Yes	Yes	No	No	No	No
Upload Python files	Yes	Yes	Yes	No	No	No	No
Upload JAR files	Yes	Yes	Yes	No	No	No	No
Upload TXT files	Yes	Yes	Yes	No	No	No	No
Upload files as Archive resources	Yes	Yes	Yes	No	No	No	No

Workflow development

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Run or stop workflows	Yes	Yes	Yes	No	No	No	No
Save workflows	Yes	Yes	Yes	No	No	No	No
View workflows	Yes	Yes	Yes	Yes	Yes	Yes	No
Commit the code of nodes	Yes	Yes	Yes	No	No	No	No
Modify workflows	Yes	Yes	Yes	No	No	No	No
View the workflow list	Yes	Yes	Yes	Yes	Yes	Yes	No
Change the workflow owner	Yes	Yes	No	No	No	No	No
View the code of nodes	Yes	Yes	Yes	No	No	No	No
Delete workflows	Yes	Yes	Yes	No	No	No	No
Create workflows	Yes	Yes	Yes	No	No	No	No
Migrate database tables	Yes	Yes	Yes	No	No	No	No
Create folders	No	Yes	Yes	No	No	No	No
Delete folders	No	Yes	Yes	No	No	No	No
Modify folders	No	Yes	Yes	No	No	No	No
Export workflows	No	Yes	Yes	Yes	No	No	No

Workspace management

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the basic information about a workspace	Yes	Yes	Yes	Yes	No	No	No
Create baselines	Yes	Yes	Yes	No	No	No	No
Delete baselines	Yes	Yes	Yes	No	No	No	No
Edit baselines	Yes	Yes	No	No	No	No	No
Search for baselines	Yes	Yes	Yes	No	No	No	No
View baselines	Yes	Yes	No	No	No	No	No

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Test connectivity	Yes	Yes	No	No	No	No	No
Create connections	Yes	Yes	No	No	No	No	No
Delete connections	Yes	Yes	No	No	No	No	No
Edit connections	Yes	Yes	No	No	No	No	No
Search for connections	Yes	Yes	No	No	No	No	No
View the connections configured for a workspace	Yes	Yes	Yes	Yes	Yes	No	No
Enable scheduling	Yes	Yes	No	No	No	No	No
View the settings of scheduling properties of nodes	Yes	Yes	No	No	No	No	No
Add workspace members	Yes	Yes	No	No	No	No	No
Change the roles of workspace members	Yes	Yes	No	No	No	No	No
View the members of a workspace	Yes	Yes	No	No	No	No	No
Remove workspace members	Yes	Yes	No	No	No	No	No
Search for workspace members	Yes	Yes	No	No	No	No	No
Modify the configurations of compute engines	Yes	Yes	No	No	No	No	No
View the configurations of compute engines	Yes	Yes	No	No	No	No	No
Query the members of within the tenant	Yes	Yes	No	No	No	No	No
Modify the basic information about a workspace	Yes	Yes	No	No	No	No	No
View the security policies of compute engines	Yes	Yes	No	No	No	No	No
Modify the security policies of compute engines	Yes	Yes	No	No	No	No	No

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Query the resource groups that are bound to a workspace	Yes	Yes	Yes	No	No	No	No
Query the servers in a resource group	No	Yes	No	No	No	No	No
Delete resource groups	No	Yes	No	No	No	No	No
Remove servers from a resource group	No	Yes	No	No	No	No	No
Configure resource groups for sync nodes	Yes	Yes	Yes	Yes	Yes	Yes	No
Bind multiple resource groups to a workspace	No	Yes	No	No	No	No	No
Add servers to a resource group	No	Yes	No	No	No	No	No
Create resource groups for a workspace	No	Yes	No	No	No	No	No
Query the projects to which a resource group is bound	No	Yes	No	No	No	No	No
Create connections	Yes	Yes	No	No	No	No	No
Edit connections	Yes	Yes	No	No	No	No	No
Share connections	Yes	Yes	No	No	No	No	No
Delete connections	Yes	Yes	No	No	No	No	No
Initialize servers in a resource group	No	Yes	No	No	No	No	No
View the connections configured for a workspace	Yes	Yes	Yes	Yes	Yes	No	No
Test connectivity	Yes	Yes	No	No	No	No	No
Search for connections	Yes	Yes	No	No	No	No	No
Update the server status and slots of a resource group	No	Yes	No	No	No	No	No
Create real-time sync nodes	Yes	Yes	Yes	No	No	No	No

Workflow O&M

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the DAG	Yes	Yes	Yes	Yes	No	No	No
Go to the DataStudio page	Yes	Yes	Yes	No	No	No	No
View the DAG of an instance	Yes	Yes	Yes	Yes	No	No	No
View ancestor and descendant nodes in the DAG	Yes	Yes	Yes	Yes	No	No	No
View the list of workflows	Yes	Yes	Yes	Yes	No	No	No
View the operations logs of workflows	Yes	Yes	Yes	Yes	No	No	No
Perform smoke tests	Yes	Yes	Yes	Yes	No	No	No
Generate retroactive data for nodes	Yes	Yes	Yes	Yes	No	No	No
Change the owner of a node	Yes	Yes	Yes	Yes	No	No	No
Unpublish workflows	Yes	Yes	Yes	Yes	No	No	No
View the details of workflows	Yes	Yes	Yes	Yes	No	No	No
View ancestor and descendant instances of an instance in the DAG	Yes	Yes	Yes	Yes	No	No	No
Pause instances	Yes	Yes	Yes	Yes	No	No	No
Restore instances	Yes	Yes	Yes	Yes	No	No	No
Terminate an instance	Yes	Yes	Yes	Yes	No	No	No
Terminate multiple instance at a time	Yes	Yes	No	Yes	No	No	No
View the list of instances	Yes	Yes	Yes	Yes	No	No	No
View operational logs	Yes	Yes	Yes	Yes	No	No	No
Rerun an instance	Yes	Yes	Yes	Yes	No	No	No
Rerun multiple instances at a time	Yes	Yes	No	Yes	No	No	No
Rerun multiple instances at a time	Yes	Yes	Yes	Yes	No	No	No
Search for instances	Yes	Yes	Yes	Yes	No	No	No

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Set the status of an instance to Successful	Yes	Yes	Yes	Yes	No	No	No
View the lineage of nodes	Yes	Yes	Yes	Yes	Yes	No	No
View node details	Yes	Yes	Yes	Yes	Yes	No	No
View the operations logs of nodes	Yes	Yes	Yes	Yes	Yes	No	No
Freeze and pause nodes	Yes	Yes	Yes	Yes	Yes	No	No
Unfreeze and resume nodes	Yes	Yes	Yes	Yes	Yes	No	No
Change the baseline for nodes	Yes	Yes	Yes	Yes	Yes	No	No
Resume instances	Yes	Yes	Yes	Yes	Yes	No	No
Delete instance dependencies	Yes	Yes	No	Yes	No	No	No
Change the running priority of instances	Yes	Yes	No	Yes	No	No	No
Forcibly rerun instances	Yes	Yes	No	Yes	No	No	No
View the lineage of instances	Yes	Yes	Yes	Yes	Yes	No	No
View instance details	Yes	Yes	Yes	Yes	Yes	No	No
View runtime logs of instances	Yes	Yes	Yes	Yes	Yes	No	No
View the baselines affected by instances	Yes	Yes	Yes	Yes	Yes	No	No
Unpublish nodes	Yes	Yes	No	Yes	No	No	No

Node maintenance

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Change the baseline for a node	Yes	Yes	Yes	Yes	No	No	No
Change the baseline for multiple nodes at a time	Yes	Yes	No	Yes	No	No	No

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the code of a node	Yes	Yes	Yes	Yes	No	No	No
Change the owner of a node	Yes	Yes	Yes	Yes	No	No	No
Change the owner of multiple nodes at a time	Yes	Yes	No	Yes	No	No	No
Change the resource group for a node	Yes	Yes	Yes	Yes	No	No	No
Change the resource group for multiple nodes at a time	Yes	Yes	Yes	Yes	No	No	No
Perform smoke tests	Yes	Yes	Yes	Yes	No	No	No
Generate retroactive data for nodes	Yes	Yes	Yes	Yes	No	No	No
Delete instance dependencies	Yes	Yes	No	Yes	No	No	No
Pause instances	Yes	Yes	Yes	Yes	No	No	No
Resume instances	Yes	Yes	Yes	Yes	No	No	No
Refresh the attribute information about instances	Yes	Yes	Yes	Yes	No	No	No
Terminate an instance	Yes	Yes	Yes	Yes	No	No	No
Terminate multiple instance at a time	Yes	Yes	No	Yes	No	No	No
Change the running priority of instances	Yes	Yes	Yes	Yes	No	No	No
Refresh the dependencies of instances	Yes	Yes	Yes	Yes	No	No	No
Rerun an instance	Yes	Yes	Yes	Yes	No	No	No
Rerun multiple instances at a time	Yes	Yes	No	Yes	No	No	No
Set the status of an instance to Successful	Yes	Yes	Yes	Yes	No	No	No
Create data quality rules	Yes	Yes	Yes	Yes	No	No	No
Delete data quality rules	Yes	Yes	Yes	Yes	No	No	No

Dashboard

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the number of baselines in the Overtime state	Yes	Yes	Yes	Yes	No	No	No
Remove a record from the dashboard	Yes	Yes	Yes	Yes	No	No	No
View the distribution of nodes by status	Yes	Yes	Yes	Yes	No	No	No
View the running information about nodes	Yes	Yes	Yes	Yes	No	No	No
View the distribution of nodes by type	Yes	Yes	Yes	Yes	No	No	No

Baseline checks

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View the metrics of baselines	Yes	Yes	Yes	Yes	No	No	No
View baselines	Yes	Yes	Yes	Yes	No	No	No

Monitoring and alerts

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
View notification messages	Yes	Yes	Yes	Yes	No	No	No
Disable an alert	Yes	Yes	Yes	Yes	No	No	No
Disable multiple alerts at a time	Yes	Yes	No	No	No	No	No
Enable or disable call notifications	Yes	Yes	Yes	Yes	No	No	No
Create custom notification rules	Yes	Yes	Yes	Yes	No	No	No
Delete custom notification rules	Yes	Yes	Yes	No	No	No	

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Edit custom notification rules	Yes	Yes	Yes	Yes	No	No	No
View custom notification rules	Yes	Yes	Yes	Yes	No	No	No
View all events	Yes	Yes	Yes	Yes	No	No	No
View event details	Yes	Yes	Yes	Yes	No	No	No
View details of personal events	Yes	Yes	Yes	Yes	No	No	No

Data integration

Permission	Workspace owner	Workspace administrator	Developer	Administration expert	Deployment expert	Visitor	Security expert
Resource consumption monitoring menu	Yes	Yes	No	No	No	No	No
View nodes	Yes	Yes	Yes	No	No	No	No
Edit nodes	Yes	Yes	Yes	No	No	No	No
Monitor resource consumption	Yes	Yes	No	No	No	No	No
Delete nodes	Yes	Yes	Yes	No	No	No	No
Migrate database tables	Yes	Yes	No	No	No	No	No

2.19.4. Manage connections

Connections are used to configure readers and writers during data integration. On the Data Source page of a workspace, you can view and add connections.

Procedure

1. Log on to the DataWorks console.
2. On the **DataStudio** page, click  in the upper-right corner.
3. In the left-side navigation pane, click **Data Source**. On the **Data Source** page, you can filter connections by conditions such as **Connect To** and **Connection Name**.

Click **Add Connection** in the upper-right corner to add a connection. For more information, see [Data sources](#).

3. Realtime Compute

3.1. What is Realtime Compute?

Realtime Compute is a big data processing platform that analyzes streaming data in real time based on Apsara Stack. You can create streaming data analysis and computing jobs by using Alibaba Cloud Flink SQL. When you use Flink SQL, you do not need to develop the underlying logic for streaming data processing.

As the demands for high data timeliness and operability increase, software systems need to process more data in less time. In traditional models for big data processing, online transaction processing (OLTP) and offline data analysis are separately performed at different times.

Realtime Compute is designed to respond to the increasing demand for the timeliness of data processing. The business value of data decreases as time passes by. Therefore, data must be computed and processed as soon as possible after it is generated. Traditional models for big data processing follow the scheduled processing mode, which accumulates and processes data in a computing cycle that can last hours or even days. These models cannot satisfy the growing demand for real-time big data processing. Batch processing cannot meet the business requirements in scenarios in which an extremely low processing delay is required. These scenarios include real-time big data analysis, early warning and risk control, real-time forecasting, and financial transactions. Realtime Compute enables real-time processing over data streams. With Realtime Compute, you can shorten the data processing delay, easily implement real-time computational logic, and greatly reduce computing costs. This helps you meet business requirements for real-time processing of big data.

Streaming data

Big data can be viewed as a series of discrete events. These discrete events form event streams or data streams along a timeline. Streaming data is continuously generated from thousands of data sources and is typically sent in data records. Streaming data has a smaller scale than offline data. Each type of data is produced as a stream of events. Streaming data includes a wide variety of data, such as the log files generated by your mobile or web applications, online purchases, in-game player activities, information from social networks, financial trade centers, geospatial services, and telemetry data from connected devices in data centers.

Realtime Compute has the following advantages:

- Real-time and unbounded data streams

Realtime Compute processes data streams in real time. Streaming data is continuously generated from data sources and is subscribed and consumed in chronological order. Data streams continuously flow into the Realtime Compute system. For example, when Realtime Compute processes data streams from website visit logs, the log data streams continuously enter the Realtime Compute system as long as the website is online. In Realtime Compute, unbounded data streams are processed in real time.

- Continuous and efficient computing

Realtime Compute is an event-driven system in which unbounded event or data streams continuously trigger real-time computations. Each streaming data record triggers a computational task. Realtime Compute performs continuous and real-time computations over data streams.

- Real-time integration of streaming data

Realtime Compute writes the computing result of each streaming data record into the target data store in real time. For example, the system directly writes the data of a computed report to an ApsaraDB for RDS instance for report display. Realtime Compute continuously writes the results into the target data store in real time. Therefore, Realtime Compute can be viewed as a data source that generates data streams for the target data store.

3.2. Quick start

3.2.1. Log on to the Realtime Compute console

This topic describes how to log on to the Realtime Compute console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, move the pointer over Products and click **Realtime Compute**.
5. Specify **Organization** and **Region**.
6. Click **Blink**.

3.2.2. Real-time security monitoring

3.2.2.1. Overview

With the wide application of digital technologies, every industry is facing the ever-increasing demand for data security, especially for real-time monitoring and alerting. To monitor streaming data and report alerts in real time, you need to ensure that the data is accurate and is processed instantly after it has been generated. To address these challenges, Realtime Compute allows you to perform JOIN operations on source tables of streaming data and dimension tables that include blacklists. The following sections describe a use case of real-time monitoring and alerting.

3.2.2.2. Preparations

Before proceeding with the development process in the Realtime Compute console, you must create a source table and a result table in the upstream and downstream data stores, respectively. You must also upload data to the source table.

Context

To simplify operations, this example organizes incoming streaming data based on a table: datahub_lpPlace.

datahub_lpPlace

Field name	Data type	Description
name	VARCHAR	The name
Place	VARCHAR	The place

The rds_dim dimension table is described as follows.

rds_dim

Field name	Data type	Description
name	VARCHAR	The name
Place	VARCHAR	The place

A result table named rds_lpPlace is obtained after a JOIN operation is performed on the datahub_lpPlace and rds_dim tables. This result table is described as follows.

rds_lpPlace

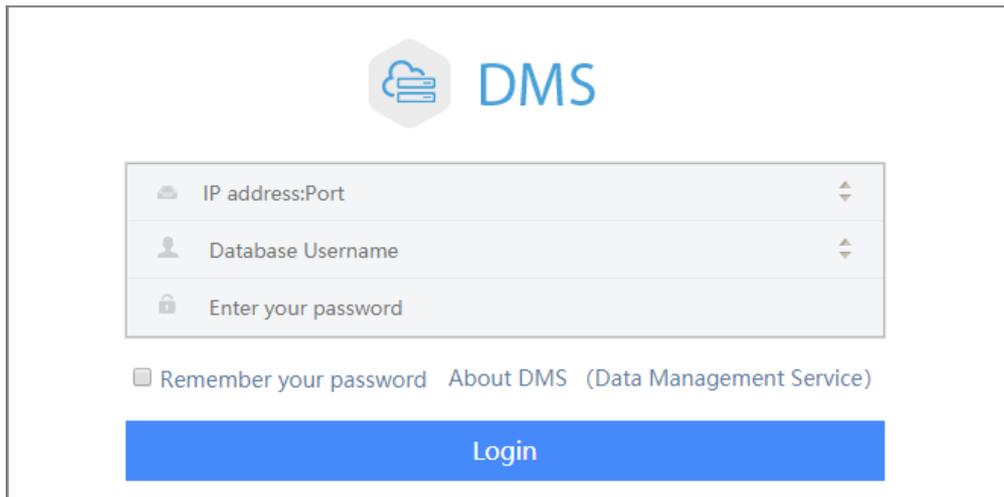
Field name	Data type	Description
name	VARCHAR	The name
Place	VARCHAR	The place

Create source and result tables

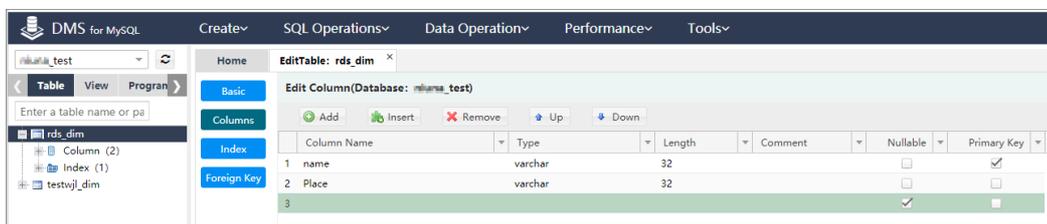
Procedure

1. Log on to the DataHub console. For more information, see the "Log on to the DataHub console" section in *DataHub User Guide*.
2. Create a project. For more information, see the "Create projects" section in *DataHub User Guide*.
3. On the page that shows the project details, create a topic. The following figure shows the schema of the source table. For more information, see the "Create a topic" section in *DataHub User Guide*.
4. Create an ApsaraDB for RDS instance. For more information, see the "Create an instance" section in *ApsaraDB for RDS User Guide*.
5. On the **Instances** page of the ApsaraDB for RDS console, click the target instance name.
6. On the **Basic Information** page, click **Log On to DB**.
7. Log on to the RDS database, and create a result table in the database.

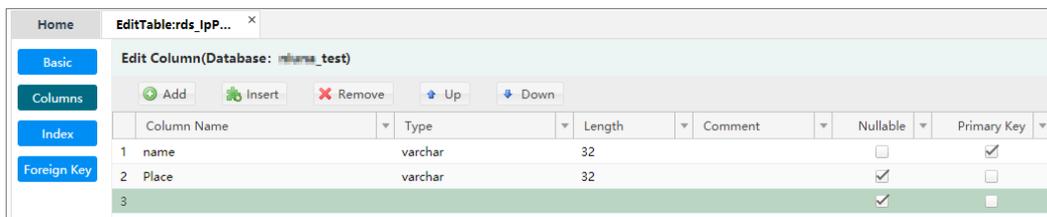
Log on to the database



Dimension table schema



Result table schema



8. Upload data to DataHub.

Log on to the **DataHub** console. In the left-side navigation pane, click **Data Acquisition**, and then click **Upload File**. On the page that appears, double-click the project, click the target DataHub topic, and then click **Select File** to upload data. For more information, see the "Upload local files" section in *DataHub User Guide*.

9. Write data into the rds_dim dimension table.

- i. Log on to the RDS database. In the top navigation bar, choose **SQL Operations > SQL Window**.
- ii. On the **SQL Window** tab, enter the following code.

```
insert into 'rds_dim'('name','Place') VALUES('test01','beijing')
```

- iii. On the **SQL Window** tab, click **execute**.

3.2.2.3. Develop a job

After you create source and result tables in external data stores, you must register the data stores in the Realtime Compute console and create references to the source and result tables. After the data stores are registered, you can proceed with the development process.

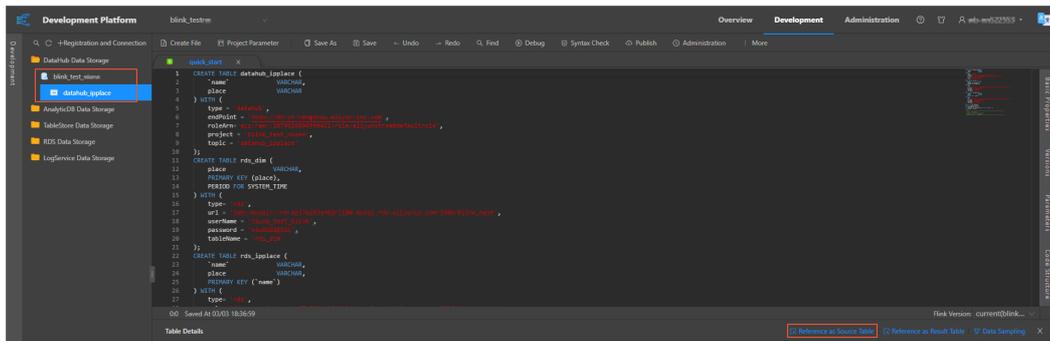
Procedure

1. [Log on to the Realtime Compute console](#) to go to the homepage of Realtime Compute.
2. In the top navigation bar, click Development.
3. On the top of the page that appears, click the **Create File** icon. Parameter description

Parameter	Description
File Name	The name of the file. The name must be 3 to 64 characters in length and can contain lowercase letters, digits, and underscores (_). It must start with a lowercase letter.
File Type	The type of the file. Valid values: FLINK_STREAM / SQL and FLINK_STREAM / DATASTREAM.
Storage Path	The folder where the job SQL file is located. You can click the folder icon on the right side of an existing folder and create a subfolder.

4. Select DataHub as the data store for a source table. You can use the DataHub parameter settings and schema information that are automatically generated on the Data Storage tab. For more information, see [Register a DataHub project](#).

Use DataHub as the data store for a source table



5. Double-click a DataHub project in the **DataHub Data Storage** folder.
6. On the top of the section that appears, click **Reference as Source Table**.
7. Create a reference to the ApsaraDB for RDS dimension table. You must specify the required parameters and the table schema. Dimension tables cannot be registered on the Storage tab. The following sample code demonstrates how to create an ApsaraDB for RDS dimension table:

```
create table rds_input (
  place varchar,
  `name` varchar,
  primary key (place),
  period for system_time
) with (
  type = 'rds',
  url = 'yourURL',
  tableName = 'yourTableName',
  password = 'yourPassword'
);
```

8. Select RDS as the data store for the result table. You can use the RDS parameter settings and schema information that are automatically generated on the Data Storage tab. For more information, see [Register an RDS instance](#).
9. To use RDS to store the result table, double-click the RDS Data Storage folder, double-click the target database, and then double-click the target table. On the top of the section that appears, click

Reference as Result Table.

10. Edit SQL statements that implement the computing logic in the code editor.

```
INSERT INTO rds_output
SELECT
  t.`name`,
  w.place
FROM datahub_input1 as t
JOIN rds_input FOR SYSTEM_TIME AS OF PROCTIME() as w
ON t.place = w.place
```

11. Click **Debug**.
12. Click **Publish**. After the computing logic is verified during the debugging, click **Publish** on the **Development** page to publish the job SQL file. Then, you can view the job on the **Administration** page of the Realtime Compute console and manage the job in the production environment, such as starting the job.

3.2.2.4. Administration

After you create and publish the job SQL file on the **Development** page, you can manage the job on the **Administration** page. For example, you can start, suspend, terminate, publish, or unpublish the job.

Procedure

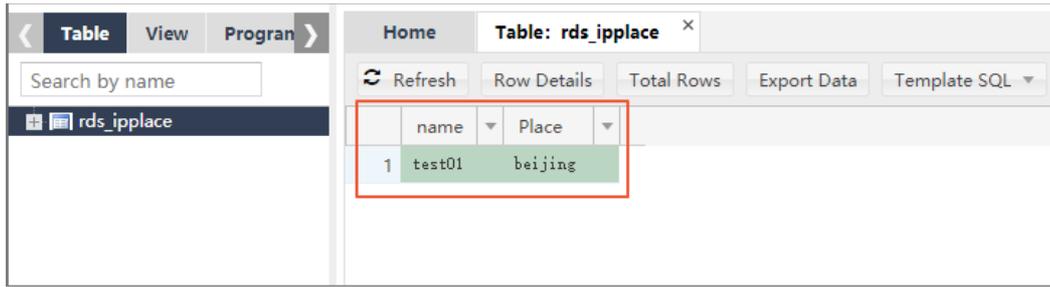
1. Go to the **Administration** page of the Realtime Compute console.
 - i. [Log on to the Realtime Compute console](#).
 - ii. In the top navigation bar, click **Administration**.
2. Click **Start** for the `bj_dim_join` job.
3. Specify the **Start Time for Reading Data** parameter. The start time for reading data is also known as the start offset. The source data store remains connected to Realtime Compute for only a short period of time. Therefore, we recommend that you specify a time that is earlier than the current time.

The **Start Time for Reading Data** parameter specifies the start time for reading data from the source data store.

Parameter	Description
Start Time for Reading Data	Specifies the start time for reading data from the source data store.
Enable Auto Upgrade	Specifies whether to enable auto upgrade.
Upgrade Time	Specifies the upgrade time range.
Offset	Specifies the range of data that is to be updated.

4. Click **OK** to start the job.
5. View the output data in the ApsaraDB for RDS data store.

View the output data in the RDS data store



3.2.3. Frequently used words

3.2.3.1. Overview

Statistical analysis of frequently used words is widely applied in diverse fields, including the analysis of frequently used words in search engines, forums, and tags. For example, you can easily view the latest and most frequently searched words in microblogging websites through real-time statistics. Statistical analysis of frequently used words is, at its core, a simple word count job. In word count jobs for streaming data, real-time processing logic is used to analyze and display frequently used words in real time.

If you are new to working with big data computing, a word count job is for you to easily get started. The word count job in big data computing is similar to a `Hello, World!` program that is often the first program that a developer learns to write. The following topics take a word count job in Realtime Compute as an example to describe how to create a word count job based on real-time processing logic. This example helps you quickly get familiar with basic Flink SQL syntax and basic operations of Realtime Compute jobs, such as creating an SQL file for a job and publishing the job.

3.2.3.2. Code development

This topic uses a word count job as an example to describe how to create a Realtime Compute job.

Prerequisites

Before creating a word count job, a source table named `stream_source` and a result table named `stream_result` are created in external data stores. The `stream_source` table includes only one column. The column is named `word` and its data type is `STRING`. The `stream_result` table includes two columns. One column is named `word` and its data type is `STRING`. The other column is named `cnt` and its data type is `BIGINT`. The two tables are registered in Realtime Compute.

1. [Log on to the Realtime Compute console.](#)
2. In the top navigation bar, click **Development**.
3. Right-click the folder where you want to store the job file.
4. Click **Create File**, and the **Create File** page appears.
5. On the page that appears, set the following parameters:
 - o File Name: Set the value to `wordcount`.
 - o File Type: Set the value to `FLINK_STREAM/SQL`.
 - o Storage Path: Keep the default setting.
6. Enter the following code in the code editor.

Note In the SQL statements for the word count job, the `STRING` data type for the referenced table must be declared as the `VARCHAR` data type.

```
create table stream_source (word varchar);
create table stream_result (word varchar, cnt bigint);
insert into
  stream_result
select
  t.word,
  count (1)
from
  stream_source t
group by
  t.word;
```

The following section explains the SQL code.

Line 1 creates a reference to the `stream_source` source table.

Note Streaming data continuously enters Realtime Compute and triggers stream processing procedures. Each streaming data record or each batch of data from the `stream_source` table triggers a stream processing procedure.

Line 2 creates a reference to the `stream_result` result table. The `stream_result` table stores the computing results of the word count job.

Note Realtime Compute does not have built-in components for data storage, and the result data is stored in external data stores, such as ApsaraDB for RDS and Tablestore. This line of code creates a reference to a result table that contains the result data.

Lines 5 through 11 implement the computing logic: Realtime Compute reads data from the `stream_source` table and counts how often words occur based on incoming data records.

Note Flink SQL supports most standard SQL statements. This allows you to easily and cost-effectively adopt Realtime Compute for stream processing.

The method of performing a word count job for stream processing is similar to that for batch processing. The word count job for stream processing continuously processes unbounded data streams until the job is terminated.

3.2.3.3. Code debugging

Realtime Compute provides a powerful debugging feature to verify SQL statements. You can debug Realtime Compute jobs by simulating data stores where streaming data, static data, and result data are stored.

- Note**
- To avoid negative impacts on online data stores, Realtime Compute is not allowed to read data from these data stores during the debugging process. Before debugging, you must prepare test data for input tables.
 - The outputs of INSERT operations are exported only to local screens. This does not affect online systems.

Debugging method

1. On the top of the **Development** page, click **Debug**.

2. On the page that appears, click **Download Template** and edit the template based on your debugging rules.

- Note** The file that is uploaded for debugging must meet the following requirements:
- The file size cannot exceed 1 MB, and the file contains maximum of 1,000 records.
 - The file must use UTF-8 encoding.
 - Commas (,) cannot be used in test data, because the file uses the comma-separated values (CSV) format.
 - Numeric values can be displayed only in the general format, and cannot be displayed in the scientific notation format.

3. Click **Upload** to upload the file.
4. Click **OK**.
5. View the debugging result in the output window.

Sample file for debugging the word count job

- Note** The file for debugging is in the CSV format. We recommend that you use the following software applications to open and modify the template:
- Excel for Windows users.
 - Vim or Sublime Text for MacOS users. We do not recommend that you use Number because it adds unnecessary fields when you modify CSV files.

Sample file for debugging the word count job



	A	B	C	D	E	F
1	word(String)					
2	aliyun					
3	aliyun					
4	aliyun					
5						
6						

Test data

You can download [test data](#) and upload the data on the **Debug File** page.

- Note** The *test data for statistical analysis of frequently used words* is unavailable for download in a PDF file. You can contact system administrators to download the test data.

View the debugging result

Real-time computing is triggered by data streams. Each data record from the `stream_source` table triggers a stream processing procedure. After each procedure is complete, a computing result is exported. The test file contains three data records. After each data record reaches Realtime Compute, a stream processing procedure is triggered. Therefore, a total of three data records are displayed on the screen. Realtime Compute uses the following computing logic:

- The first data record (aliyun) reaches Realtime Compute. This is the first time that the system has detected the word "aliyun." Therefore, the computing result is `<aliyun, 1>`, which is displayed on the screen.

- The second data record (aliyun) reaches Realtime Compute. The system detects an existing record of `<aliyun, 1>` and increases the value by one. Therefore, the computing result is `<aliyun, 2>`, which is displayed on the screen.
- The third data record (aliyun) reaches Realtime Compute. The system detects an existing record of `<aliyun, 2>` and increases the value by one. Therefore, the computing result is `<aliyun, 3>`, which is displayed on the screen.

The third computing result `<aliyun, 3>` is considered as the final output of the debugging. Another sample of **test data** is provided for you to test the debugging feature. You can use different samples of test data and view the debugging outputs.

3.2.3.4. Administration

After the SQL file has been verified, you can publish the SQL file for the job on the **Administration** page of Realtime Compute. Then, you can start the job. The job runs on a Realtime Compute cluster.

Procedure

1. On the **Development** page, click **Publish**. The **Publish New Version** dialog box appears.
2. In the **Resource Configuration** step, click **Next**.
3. In the **Check** step, click **Next**.
4. In the **Publish File** step, click **Publish**.
5. On the **Administration** page, view the published word count job.
6. Click **Start** in the **Actions** column of the word count job. The **Start** dialog box appears.
7. Specify **Start Time of Reading Data** and click **OK**. Then, the job runs on a Realtime Compute cluster.

Result

After the job is started, click the job name. The **Overview** page appears.

FAQ

Q: Why does the word count job have no input or output while it is running on the distributed compute clusters of Realtime Compute?

A: When you created the `my_source` and `my_result` tables, you did not specify the data storage type of the referenced data source. In this scenario, the source table is considered to be a random table of strings or digits, and the result table is considered to be discarded data.

3.2.4. Big screen service for the Tmall Double Eleven Global Shopping Festival

3.2.4.1. Overview

During Double 11, a big screen shows the total sales volume of Alibaba Group in real time. The big screen service is a highlight for the shopping festival.

Stream processing for the big screen service was previously based on Apache Storm that is an open source distributed real-time computation system. The Apache Storm-based development process took around one month. The application of Flink SQL reduces the period for the development of the big screen service to three days. The underlying layer of Realtime Compute removes the Apache Storm modules that are designed for execution optimization and troubleshooting. This achieves a higher processing efficiency for Realtime Compute jobs.

3.2.4.2. Scenario description

The streaming data input for the Tmall big screen service is the transaction data from the Tmall platform. The incoming transaction data is organized based on a two-dimensional table: `tmall_trade_detail`.

Field	Type	Description
tid	BIGINT	The order ID.
buyer_uid	BIGINT	The buyer ID.
seller_uid	BIGINT	The seller ID.
gmtdate	TIMESTAMP	The time when the order is completed.
payment	DOUBLE	The order amount.

Realtime Compute calculates two metrics based on the preceding table: the total number of orders and the total order amount up to the current time. The two metrics are written to an online RDS system and displayed on a big screen in real time. The online RDS system is used to store the result table: `tmall_trade_state`.

Field	Type	Description
gmtdate	VARCHAR(16)	The date when the order is completed.
trade_count	BIGINT	The total number of orders.
trade_sum	DOUBLE	The total order amount.

The following topics describe how to build an end-to-end solution for the Tmall big screen service in around 10 minutes.

3.2.4.3. Preparations

Before editing Flink SQL statements for a Realtime Compute job, you must register data stores for source tables and result tables in Realtime Compute. This topic uses DataHub as an example to describe how to register a data store in Realtime Compute.

Create a DataHub topic

1. Log on to the DataHub console. For more information, see the "Log on to the DataHub console" section in *DataHub User Guide*.
2. If one or more projects have been created, click **View** in the **Actions** column for the target project. If no projects have been created, click **Create Project** to create a project and click **View** in the **Actions** column.
3. On the page that appears, click **Create Topic**.
4. Configure the topic based on the schema of the `tmall_trade_state` RDS table in the "Scenario description" section.

After performing these steps, you can edit Flink SQL statements for the Realtime Compute job.

Upload data to DataHub

You can upload data to the DataHub topic that you have created. To do this, follow these steps. Log on to the **DataHub** console. You can upload data to the DataHub topic that you have created. To do this, follow these steps:

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Data Acquisition**.
3. Click **Upload File**.
4. On the page that appears, double-click the target project and click the target DataHub topic.
5. Click **Select File** to select a file.
6. Click **Upload**.

To simplify the test procedure, you can use the [test data about the Double 11](#). You can download the data and then upload it to the DataHub topic for data collection.

3.2.4.4. Register a data store

The data store registration feature of Realtime Compute allows you to easily register DataHub topics, create tables, and reference data stores. To register a data store, follow these steps:

Procedure

1. [Log on to the Realtime Compute console](#).
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click the **Storage** tab.
4. Click the **DataHub Data Storage** folder.
5. On the top of the page, click **+Registration and Connection**.
6. Register a DataHub project in Realtime Compute. For more information about parameter settings, see [Register a DataHub project](#).

If you use ApsaraDB for RDS for MySQL to store the result data for data visualization, you must register an RDS data store in Realtime Compute. For more information, see [Register an RDS instance](#).

3.2.4.5. Development

After the data has been collected to Realtime Compute, you can continue to edit Flink SQL statements.

1. Create a reference to the source.

To create references to the DataHub source table and RDS result table, click **Data Storage** in the left-side navigation pane of the **Development** page in the Realtime Compute console, and perform the following operations:

- Find the target DataHub topic, and click **Reference as Source Table**. Realtime Compute automatically parses the schema of the topic and adds the corresponding SQL statements to the **Development** page.
- Find the target RDS table, and click **Reference as Result Table**. Realtime Compute automatically parses the schema of the table and adds the corresponding SQL statements to the **Development** page.

2. Edit Flink SQL statements.

If you have created the DataHub topic and RDS table as described in the previous topics, the Flink SQL code for the `tmall_d11` job can be executed directly. Otherwise, change the names of the DataHub topic and RDS table based on the topic and table that you have created. The sample code is as follows:

```
replace into tmall_trade_state
select
  from_unixtime(FLOOR(tmall_trade_detail.gmtime/1000), 'yyyy-MM-dd') as gmt_date,
  count(tid) as trade_count,
  sum(payment) as trade_sum
from
  tmall_trade_detail
group by
  from_unixtime(FLOOR(tmall_trade_detail.gmtime/1000), 'yyyy-MM-dd');
```

 **Note** You can modify the information about tables and fields as required.

3. Debug the Flink SQL code.

The [data during the Double 11 Shopping Festival](#) is available for testing. To debug the code, download the test data and upload the data on the **Development** page for debugging.

4. Publish the SQL file for the tmall_d11 job.

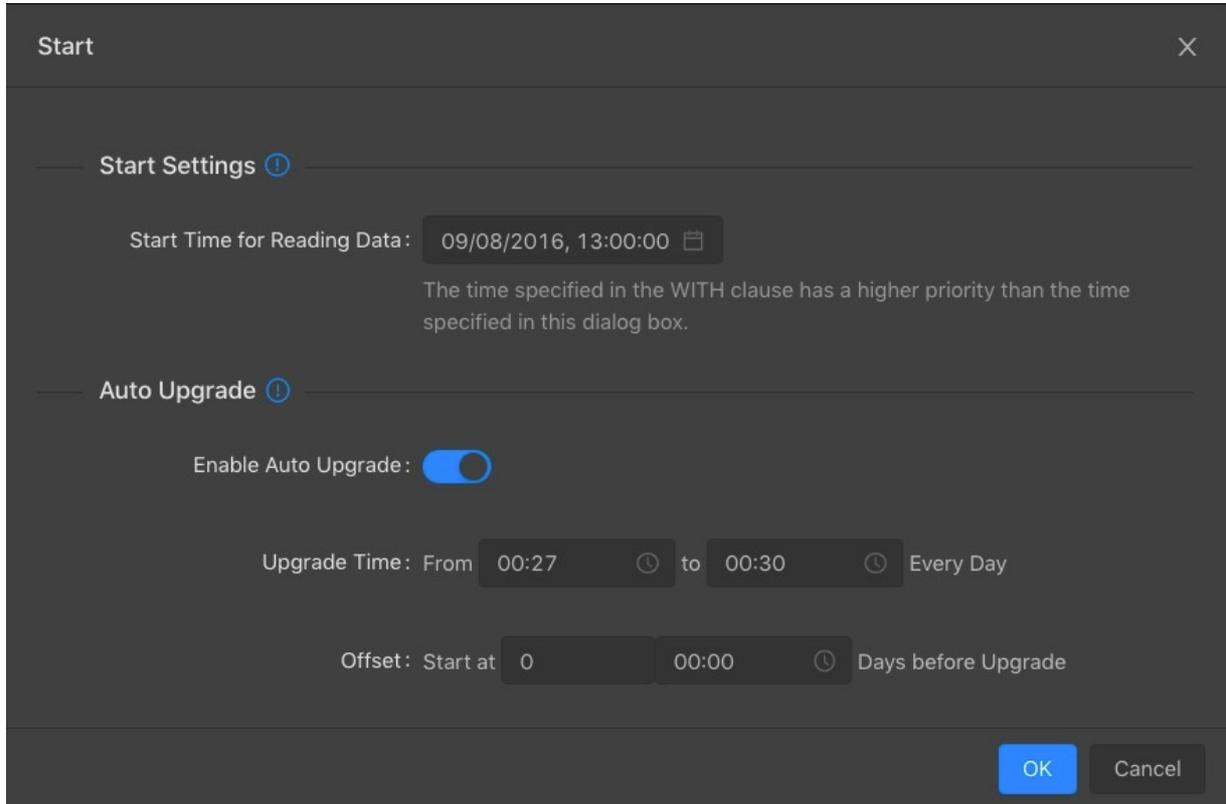
After the computational logic has been verified in the debugging phase, click **Publish** on the **Development** page to publish the SQL file for the tmall_d11 job. Then, you can view the tmall_d11 job on the **Administration** page of the Realtime Compute console, and manage the job in the production environment, such as starting the job.

3.2.4.6. Administration

On the **Administration** page, you can click **Start** in the **Actions** column and specify the parameters on the page that appears to start a stream processing job, for example, the tmall_d11 job.

 **Note** After you click **Start**, a dialog box is displayed. In the dialog box, you can specify the start time for reading data from the source data store.

The specified start time must be earlier than the file upload time. For example, the start time can be one hour earlier than the file upload time. In the Double 11 use case, the current time is 14:10, and 10 minutes have elapsed since the source data is uploaded. Therefore, the start time is set to 13:00.



The screenshot shows a 'Start' dialog box with the following settings:

- Start Settings**
 - Start Time for Reading Data: 09/08/2016, 13:00:00
 - Note: The time specified in the WITH clause has a higher priority than the time specified in this dialog box.
- Auto Upgrade**
 - Enable Auto Upgrade:
 - Upgrade Time: From 00:27 to 00:30 Every Day
 - Offset: Start at 0 00:00 Days before Upgrade

Buttons: OK, Cancel

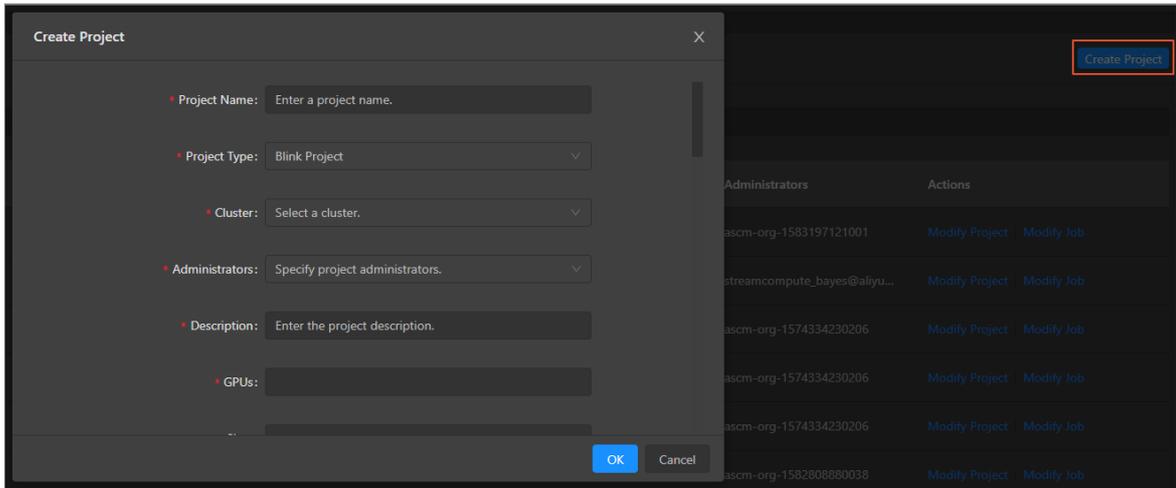
You can check the result data in the ApsaraDB for RDS data store after the job runs as expected. In the result table, five transactions and a turnover of CNY 500 are displayed. This is consistent with the source data for testing. In this way, an end-to-end verification is performed to check the SQL code.

3.3. Project management

This topic describes how to create and search for a project.

Create a project

1. [Log on to the Realtime Compute console.](#)
2. In the top navigation bar, move the pointer over your profile picture and click **Project Management**.
3. In the upper-right corner of the **Projects** page, click **Create Project**.
4. Configure the project information.



Parameter description

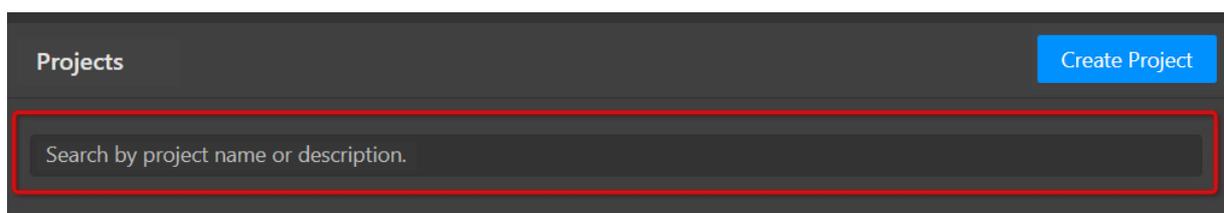
Parameter	Description
Project Name	The name of the project.
Project Type	The type of the project. Blink Project is selected by default.
Cluster	The cluster on which the jobs in the project run.
Description	The description of the project.
GPUs	The number of GPUs that are used by the project.
Slots	The number of compute units (CUs) that are used by the project. One CU is assigned with one CPU core and 4 GB memory.
Alert Methods	The methods in which alerts are sent when errors occur during job running. You can receive alerts by using text messages or TradeManager messages.
File Types	The supported file types. You can keep the default setting.
Storage Types	The supported data store types. You can keep the default setting.
Max Data Stores	The maximum number of data stores that can be registered in Realtime Compute. You can keep the default setting.
Max File Versions	The maximum number of code versions for an SQL file. You can keep the default setting.
Max Folders	The maximum number of folders that can be created in the project. You can keep the default setting.
Max Folder Levels	The maximum number of folder levels in the project. You can keep the default setting.
Max Files	The maximum number of job SQL files that can be created in the project. You can keep the default setting.
Max Resources	The maximum number of JAR files and DICT IONARY resources that can be uploaded. You can keep the default setting.

Parameter	Description
Max Referenced Resources	The maximum number of JAR files and DICTIONARY resources that can be referenced. You can keep the default setting.
Monitoring and Alerting	Specifies whether to enable the monitoring and alerting feature. You can keep the default setting.
Data Collection	Specifies whether to collect data while a job is running. You can keep the default setting.
Data Display	Specifies whether to display data. You can keep the default setting.
Metastore	Specifies whether to display metadata. You can keep the default setting.
Data Storage	Specifies whether to enable data store registration. This feature is enabled by default. You can keep the default setting.
Engine	Specifies whether to display the engine. You can keep the default setting.
Online Logs	Specifies whether to record the job running logs. This feature is enabled by default. You can keep the default setting.
Resource Management	Specifies whether resources such as JAR files can be uploaded. This feature is enabled by default. You can keep the default setting.
Switch Version	Specifies whether to enable the job version switch feature. This feature is enabled by default. You can keep the default setting.
Project Protection	Specifies whether to enable the project lock feature. You can keep the default setting.

5. Click OK.

Search for a project

On the **Projects** page, enter a keyword or full name of a project in the search box to find the project.



3.4. Data storage

3.4.1. Overview

This chapter describes data storage systems supported by Realtime Compute.

3.4.2. Overview

3.4.2.1. Overview

To facilitate data storage management, you can register data storage resources on the Realtime Compute development platform. This enables you to leverage the advantages of the one-stop Realtime Compute service. In Realtime Compute, you can manage multiple data storage systems, such as ApsaraDB for RDS, and Table Store. With this one-stop management service, you no longer have to navigate across multiple management consoles of different storage systems.

3.4.2.2. Types

Realtime Compute supports both streaming data storage and static data storage.

Streaming data storage

Streaming data storage systems provide inputs and outputs for downstream Realtime Compute jobs.

Streaming data storage

Storage system	Input	Output
DataHub	Supported	Supported
Log Service	Supported	Supported
MQ	Supported	Supported

Static data storage

Static data storage systems provide outputs for downstream Realtime Compute jobs and allow you to perform association queries.

Static data storage

Storage system	Dimension table	Output
ApsaraDB for RDS	Supported	Supported
Table Store	Supported	Supported

3.4.2.3. Registration and usage

This topic describes how to register and use external data stores in Realtime Compute.

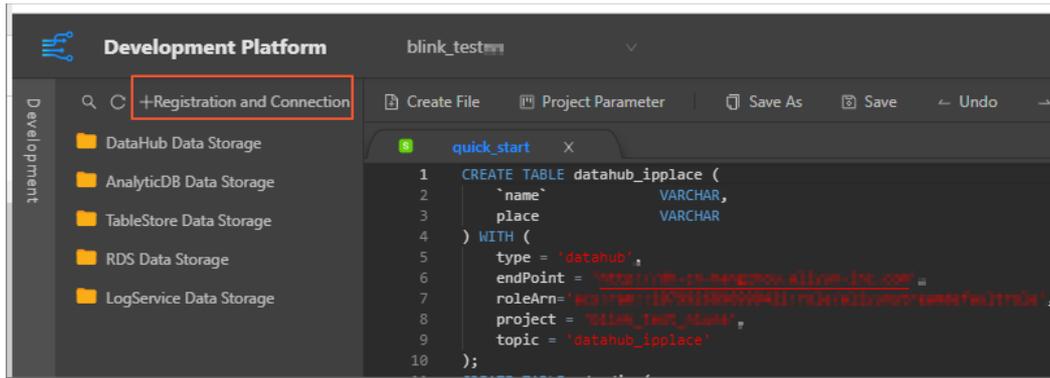
 **Note** If a job requires the use of the data stores owned by another Apsara Stack tenant account, you can write DDL statements to reference the data stores. In the DDL statements, you must specify the AccessKey ID and AccessKey secret of the account. In this scenario, you cannot use the codeless UI to manage the data stores.

Register a data store

To register a data store, follow these steps:

1. [Log on to the Realtime Compute console](#).
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane of the page that appears, click the **Storage** tab. Select the folder for the data store that you want to register. Then, click **+Registration and Connection**.

Register a data store



4. In the dialog box that appears, configure the required parameters and click OK.

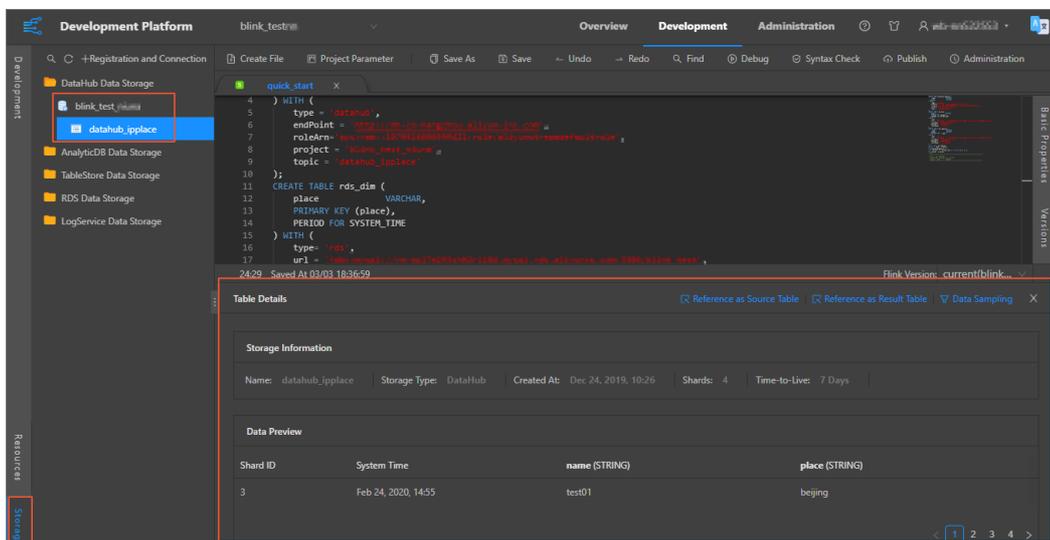
Note After you enable the data store registration feature, you can only register data stores that are owned by your organization.

Preview data from a data store

Realtime Compute provides the data preview feature for each registered data store. To preview data, click the **Storage** tab and double-click the folder of the target data store in the left-side navigation pane. The following example shows how to preview data from a DataHub data store.

1. Log on to the Realtime Compute console. In the top-navigation bar, click **Development**. On the page that appears, click the **Storage** tab and double-click the **DataHub Data Storage** folder.
2. Double-click the target project and then the target topic to view the details.

Table details



Use the auto DDL generation feature

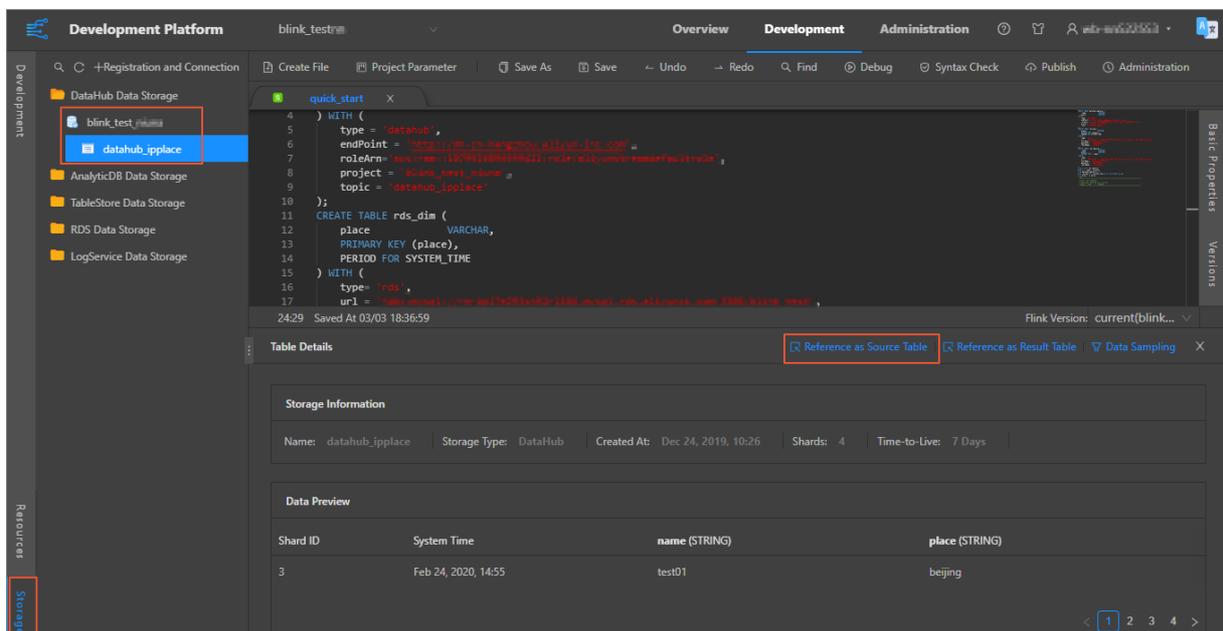
You must declare tables from external data stores before you can reference these tables. The following example shows how to reference a source table that contains streaming data:

```
CREATE TABLE in_stream( a varchar, b varchar, c timeStamp) with ( type='datahub', endPoint='http://dh-cn-hangzhou.aliyuncs.com', project='blink_test', topic='ip_count02', accessId='LTAlYtaf*****', accessKey='gUqyVwfkK2vfJl7jF90*****');
```

The field names in the table that is referenced on the Development page must be the same as those in the DataHub topic. You must declare the field data types in the code based on the field type mapping between DataHub and Realtime Compute to ensure that Realtime Compute can identify the data. Realtime Compute offers the auto DDL generation feature. The following section describes how to use this feature.

1. In the left-side navigation pane of the **Development** page, click the **Storage** tab.
2. On the **Storage** tab, navigate through cascaded folders and nodes to find the target table. Then, double-click the name of the target table.
3. In the **Table Details** pane that appears, click **Reference as Source Table**, **Reference as Result Table**, or **Reference as Dimension Table** as required. Then, you can obtain the DDL statements that are automatically generated to reference the target table.

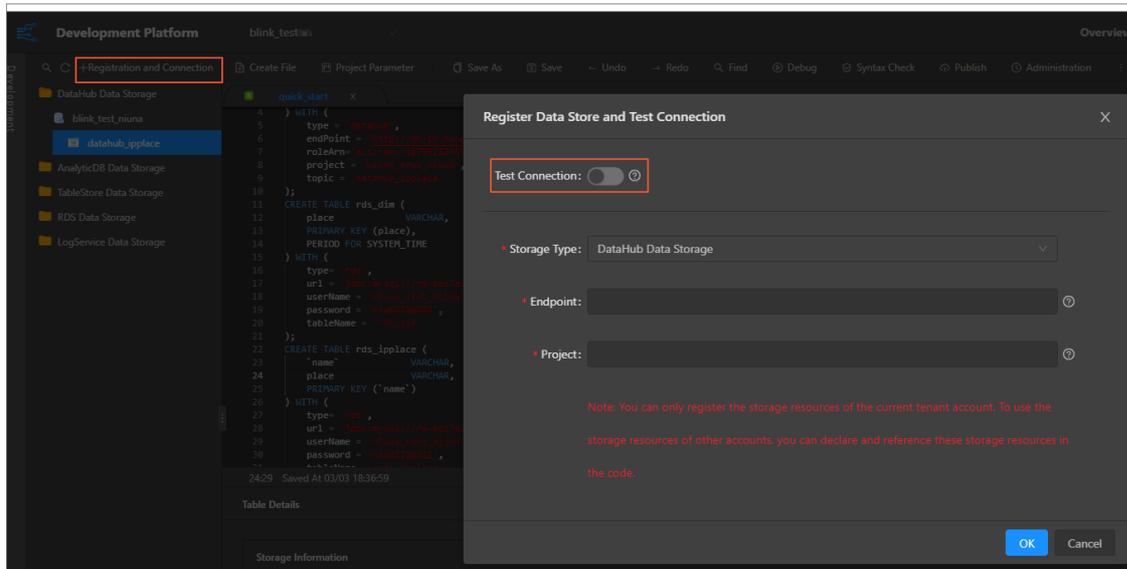
To reference a source table, log on to the Realtime Compute console and open the target SQL file on the Development page. Click the **Storage** tab, select the table for reference, and then click **Reference as Source Table**. The required DDL statements are displayed on the current page.



Test network connection

Realtime Compute offers the network connection test feature for data stores. This feature allows you to test the connection between Realtime Compute and a target data store. To enable the network connection test feature, follow these steps:

1. In the left-side navigation pane of the **Development** page, click the **Storage** tab.
2. In the upper-right corner of the **Storage** tab, click **+ Registration and Connection**.
3. In the **Register Data Store and Test Connection** dialog box, turn on **Test Connection**.



Example: Reference data stores owned by another level-1 organization

You can only register and use data stores that are owned by your level-1 organization. To use data stores that are owned by another level-1 organization, write DDL statements to create a reference to these data stores. For example, if a user from Organization A wants to use the data stores owned by Organization B, the user can enter the following DDL statements:

```
CREATE TABLE in_stream( a varchar, b varchar, c timeStamp) with ( type='datahub', endPoint='http://dh-cn-hangzhou.aliyuncs.com', project='blink_test', topic='ip_count02', accessId='AccessKey ID authorized by Organization B users', accessKey='AccessKey secret authorized by Organization B users');
```

3.4.3. Register a DataHub data store

DataHub, an Alibaba Cloud streaming data service, is a real-time data distribution platform designed to process streaming data. You can publish and subscribe to applications for streaming data in DataHub and distribute the data to other platforms. DataHub allows you to analyze streaming data and build applications based on the streaming data. Realtime Compute often uses DataHub to store source and result tables that contain streaming data.

Register a DataHub project

1. Log on to the Realtime Compute console.
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click the **Storage** tab.
4. Right-click the **DataHub Data Storage** folder and select **Register Data Store** to register a DataHub project in Realtime Compute. Parameter description

Parameter	Description
Test Connection	Specifies whether to enable the network connection test feature. Network connection tests are automatically performed on data stores that can be registered in Realtime Compute. To test the connection between Realtime Compute and data stores that cannot be registered, turn on the Test Connection switch.
Storage Type	The type of the data store. DataHub Data Storage is selected by default.

Parameter	Description
Endpoint	<p>The endpoint of DataHub. The endpoint of DataHub varies with regions. For more information about endpoints, contact your administrator.</p> <p> Note To specify this parameter for Apsara Stack, contact your Apsara Stack administrator to obtain the endpoint of DataHub.</p>
Project	<p>The name of the DataHub project.</p> <p> Note You can only register DataHub projects that are owned by your level-1 organization. For example, if DataHub Project A is owned by Organization A, users from Organization B cannot register Project A in Realtime Compute.</p>
AccessKey ID	The AccessKey ID of the current account.
AccessKey Secret	The AccessKey secret of the current account. The AccessKey secret enables Realtime Compute to access the DataHub project.

Scenarios

DataHub is a streaming data storage system that can be used to store source and result tables. However, it cannot be used to store dimension tables for Realtime Compute.

FAQ

Q: Why am I unable to register a DataHub project in Realtime Compute?

A: Realtime Compute uses a storage software development kit (SDK) to access different data stores. The Storage tab in the Realtime Compute console only helps you manage data from different data stores. You can perform the following operations to troubleshoot registration errors:

- Check whether you have created a DataHub project and have the permissions to access the project. You can log on to the DataHub console and check whether you can access the project.
- Check whether you are the project owner. You can only register DataHub projects that are owned by your level-1 organization. For example, if DataHub Project A is owned by Organization A, users from Organization B cannot register Project A in Realtime Compute.
- Check whether you have specified the correct DataHub endpoint and project name.
- Check whether you have specified a classic network endpoint for the Endpoint parameter. If you specify a VPC endpoint, the DataHub project will fail to be registered.
- Check whether you have registered the DataHub project. Realtime Compute provides a registration check mechanism to prevent duplicate registration.

Q: Why does Realtime Compute only support time-based sampling?

A: DataHub stores streaming data, and you can only specify time parameters in the API. Therefore, Realtime Compute supports only time-based sampling.

3.4.4. Register a Log Service data store

Log Service (previously known as SLS) provides an end-to-end solution for log management. You can use Log Service to easily collect, subscribe to, dump, and query large amounts of log data. Realtime Compute can integrate with Log Service to process logs. This eliminates the need of data migration.

Register a Log Service project

1. Log on to the Realtime Compute console.
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click the **Storage** tab.
4. Right-click the **LogService Data Storage** folder and select **Register Data Store** to register a Log Service project in Realtime Compute. Parameter description

Parameter	Description
Test Connection	Specifies whether to enable the network connection test feature. Network connection tests are automatically performed on data stores that can be registered in Realtime Compute. To test the connection between Realtime Compute and data stores that cannot be registered, turn on the Test Connection switch.
Storage Type	The type of the data store. LogService Data Storage is selected by default.
Endpoint	The endpoint of Log Service. The endpoint of Log Service varies with regions. <div style="background-color: #e6f2ff; padding: 5px;"> <p> Note For more information about the endpoint of Log Service, contact the Apsara Stack system administrator.</p> </div>
Project	The name of the Log Service project. <div style="background-color: #e6f2ff; padding: 5px;"> <p> Note You can only register Log Service projects that are owned by your level-1 organization. For example, if Log Service Project A is owned by Organization A, users from Organization B cannot register Project A in Realtime Compute.</p> </div>
AccessKey ID	The AccessKey ID of the current account.
AccessKey Secret	The AccessKey secret of the current account. The AccessKey secret enables Realtime Compute to access the Log Service project.

Scenarios

Log Service is a streaming data storage system that can be used to store source tables and result tables. However, it cannot be used to store dimension tables for Realtime Compute.

FAQ

- Q: Why am I unable to register a Log Service project in Realtime Compute?
 A: Realtime Compute uses a storage software development kit (SDK) to access different data stores. The Storage tab in the Realtime Compute console only helps you manage data from different data stores. You can perform the following operations to troubleshoot registration errors:
 - Check whether you have created a Log Service project and have the permissions to access the project. You can log on to the Log Service console and check whether you can access the project.
 - Check whether you are the project owner. You can only register Log Service projects that are owned by your level-1 organization. For example, if Log Service Project A is owned by Organization A, a user from Organization B cannot register Project A in Realtime Compute.

- Check whether you have specified the correct Log Service endpoint and project name.

Note The endpoint must start with `http` and cannot end with a forward slash (`/`). For example, `http://cn-hangzhou.log.aliyuncs.com` is correct, and `http://cn-hangzhou.log.aliyuncs.com/` is incorrect.

- Check whether you have registered the Log Service project. Realtime Compute provides a registration check mechanism that prevents duplicate registration.

- Q: Why does Realtime Compute support only time-based sampling?

A: Log Service stores streaming data, and you can only specify time parameters in the API. Therefore, Realtime Compute supports only time-based sampling.

Note To use the search feature of Log Service, log on to the Log Service console.

3.4.5. Register a Tablestore data store

Tablestore is a NoSQL database service that is based on the Apsara distributed system. Tablestore allows you to store and access large amounts of structured data in real time. Tablestore features massive data storage and low access delays, which makes it suitable to store dimension tables and result tables for Realtime Compute.

Register a Tablestore instance

1. [Log on to the Realtime Compute console](#).
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click the **Storage** tab.
4. Right-click **TableStore Data Storage** and select **Register Data Store**. In the dialog box that appears, register a Tablestore instance in Realtime Compute. Parameters

Parameter	Description
Test Connection	Specifies whether to enable the network connection test feature. Network connection tests are automatically performed on data stores that can be registered in Realtime Compute. To test the connection between Realtime Compute and data stores that cannot be registered, turn on the Test Connection switch.
Storage Type	The type of the data store. TableStore Data Storage is selected by default.
Endpoint	The endpoint of the Tablestore instance. You must enter the internal endpoint of the Tablestore instance. You can log on to the Tablestore console to view the internal endpoint of the instance.
Instance Name	The name of the Tablestore instance.
AccessKey ID	The AccessKey ID of the current account.
AccessKey Secret	The AccessKey secret of the current account. The AccessKey secret enables Realtime Compute to access the Tablestore instance.

3.4.6. Register an RDS data store

This topic describes how to register and use an ApsaraDB for RDS data store in Realtime Compute.

RDS overview

RDS offers a stable, reliable, and scalable online database service. Based on the Apsara distributed system and high performance SSD storage, RDS supports a wide range of engines, such as MySQL, PostgreSQL, and Postgres Plus Advanced Server (PPAS, highly compatible with Oracle). Realtime Compute supports the following RDS engines: MySQL and PostgreSQL.

The performance of Tablestore in high concurrency scenarios where large amounts of data need to be processed is higher than that of RDS. The performance of RDS is restricted by the limits of relational models. Therefore, RDS is often used to store result tables for Realtime Compute. In low concurrency scenarios where a small number of data needs to be processed, RDS can be used to store dimension tables.

Note Realtime Compute uses relational databases, such as ApsaraDB RDS for MySQL, to store result data. Distributed Relational Database Service (DRDS) and RDS connectors are used. If Realtime Compute frequently writes data into a DRDS or RDS table, deadlocks may occur. In scenarios that require high queries per second (QPS), high transactions per second (TPS), or highly concurrent write operations, we recommend that you do not use DRDS or RDS to store the result tables of Blink jobs. To prevent deadlocks, we recommend that you use Tablestore to store result tables.

Register an RDS instance

1. [Log on to the Realtime Compute console](#).
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click the **Storage** tab.
4. Right-click **RDS Data Storage**, and select **Register Data Store**. In the dialog box that appears, register an RDS instance in Realtime Compute. Parameters

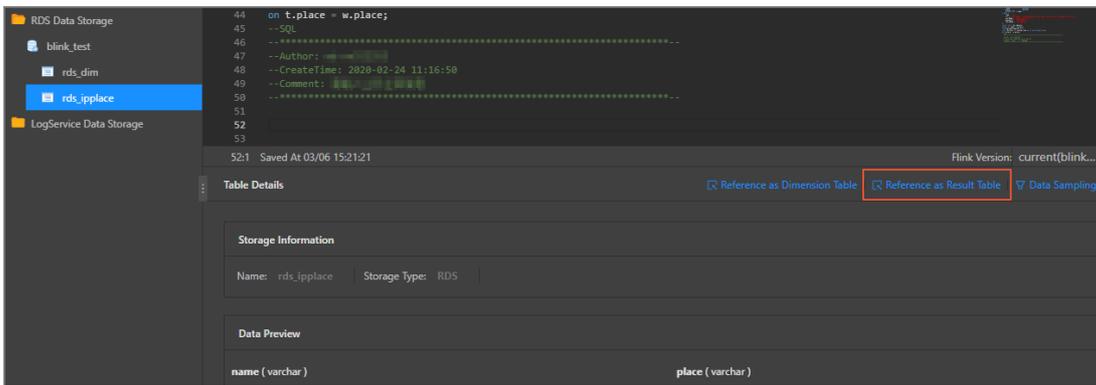
Parameter	Description
Test Connection	Specifies whether to enable the network connection test feature. Network connection tests are automatically performed on data stores that can be registered in Realtime Compute. To test the connection between Realtime Compute and data stores that cannot be registered, turn on the Test Connection switch.
Storage Type	The type of the data store. RDS Data Storage is selected by default.
URL	The connection URL of the RDS database.
DBName	<p>The name of the RDS database to be accessed by Realtime Compute.</p> <p>Note This parameter specifies the RDS database name instead of the RDS instance name.</p> <p>RDS uses whitelists for access control to ensure system security. The IP addresses of the Realtime Compute console and worker nodes must be added to RDS whitelists. Otherwise, Realtime Compute may fail to connect to RDS. For more information, see Specify whitelist settings.</p>
User Name	The username that you use to log on to the RDS database.
Password	The password that you use to log on to the RDS database.

Parameter	Description
Engine	The type of the RDS database. Valid values: <ul style="list-style-type: none"> ○ mysql ○ postgresql ○ sqlserver

Reference an RDS table as a result table

After you register an RDS data store, double-click the RDS database, double-click the RDS table that you want to reference as a result table, and then click **Reference as Result Table**.

Reference an RDS table as a result table



After you click **Reference as Result Table**, Realtime Compute automatically generates the corresponding DDL statements on the current page.

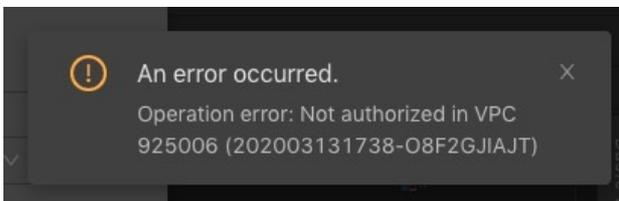
Result

```

31     place          VARCHAR,
32     PRIMARY KEY (`name`)
33 ) WITH (
34     type= 'rds',
35     url = 'dev.mysql.com:3306/blink_test?server=dev.mysql.com:3306/blink_test',
36     userName = 'blink_test_admin',
37     password = 'blink_test!!!',
38     tableName = 'rds_ipplace'
39 );
    
```

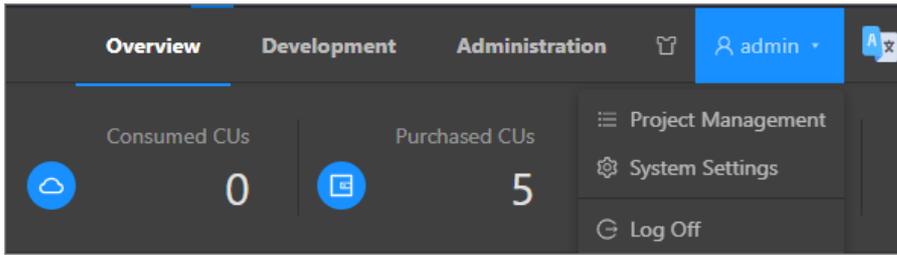
If the following error message appears, troubleshoot and rectify the fault as follows.

Error message



The error occurs because a VPC (not a classic network) was selected when you created the RDS instance. To rectify this fault, follow these steps:

1. Move the pointer over the administrator icon, as shown in the following figure.



2. Click **System Settings**.
3. In the left-side navigation pane, click **VPC Access Authorization**.
4. Click **Add Authorization**. The **Authorize StreamCompute VPC Access** page appears.

Authorization

Authorize StreamCompute VPC Access ✕

* Name:

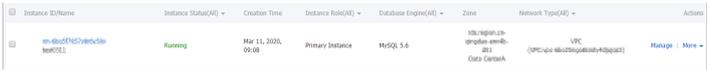
* Region: ▾

* VPC ID:

* Instance ID:

* Instance Port:

Parameters

Parameter	Description
Name	The name of the VPC.
Region	The region where RDS resides.
VPC ID	The ID of the VPC.
Instance ID	<p>The ID of the RDS instance. You can log on to the RDS console and view the instance ID.</p> <p>Instance information</p> 
Instance Port	The access port for the RDS instance. To view the internal port number, log on to the RDS console, click the target instance ID or name in the Instance ID/Name column. On the page that appears, view the internal port number in the Basic Information section.

5. Register the target RDS instance. You must specify the required parameters during the registration.

You can only register data storage resources that are owned by your level-1 organization. For example, if RDS Instance A is owned by Organization A, a user from Organization B cannot register RDS Instance A in the Realtime Compute console. To use Instance A in a stream processing job, the user from Organization B must use SQL code to create a reference to Instance A.

Note If you want to use the RDS storage resources owned by your level-1 organization, we recommend that you do not use SQL code to create a reference to these resources.

The user from Organization B must also specify the following parameters in the WITH clause based on the information of Instance A: url, userName, password, and tableName.

Configuration

```
91  
92 CREATE TABLE datahub_output (  
93   id          varchar  
94 ) WITH (  
95   type= 'rds',  
96   url = 'jdbc:mysql://  
97   userName = 'root@  
98   password = '1234567890123',  
99   tableName = 'datahub_output'  
100 );  
101
```

To use RDS storage resources by writing SQL code, the user from Organization B must specify whitelist settings.

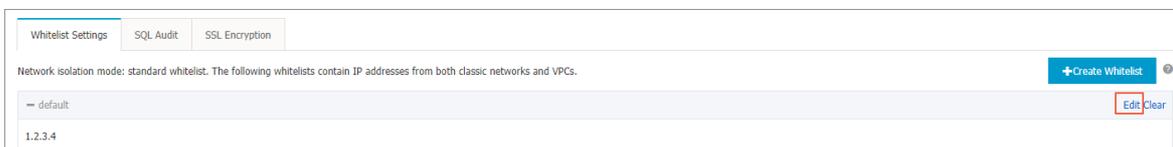
Specify whitelist settings

Some data stores use whitelists for access control to ensure high-level security. These data stores allow access only from the IP addresses that are added to the whitelists. This prevents unauthorized Apsara Stack services from accessing data in these data stores. For example, a newly created RDS database denies all access. You must add IP addresses to the RDS database whitelists to allow access to the database.

RDS can be accessed from both external and internal networks. To enable Realtime Compute to access RDS, you must add the CIDR blocks of Realtime Compute to RDS whitelists.

To do this, follow these steps:

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, click the ID of an instance in the **Instance ID/Name** column.
3. In the left-side navigation pane, click **Data Security**.
4. On the **Whitelist Settings** tab, click **Edit** corresponding to the **default** whitelist.



Note

- To connect an ECS instance to an RDS instance by using an internal endpoint, you must make sure that the two instances are in the same region and have the same network type. Otherwise, the connection fails.
- You can also click **Create Whitelist** to create a new whitelist.

5. In the **Edit Whitelist** dialog box that appears, specify the IP addresses or CIDR blocks that are used to access the instance, and then click **OK**.

Edit Whitelist

*Whitelist Name: default

*IP Addresses: 127.0.0.1

[Add Internal IP Addresses of ECS Instances](#)
You can add 999 more entries.

Specified IP address: If you specify the IP address 192.168.0.1, this IP address is allowed to access the RDS instance.
Specified CIDR block: If you specify the CIDR block 192.168.0.0/24, the IP addresses ranging from 192.168.0.1 to 192.168.0.255 are allowed to access the RDS instance.
When you add multiple IP addresses or CIDR blocks, separate them by a comma (no space after the comma), for example, 192.168.0.1,192.168.0.0/24.

New whitelist entries take effect in 1 minute.

OK Cancel

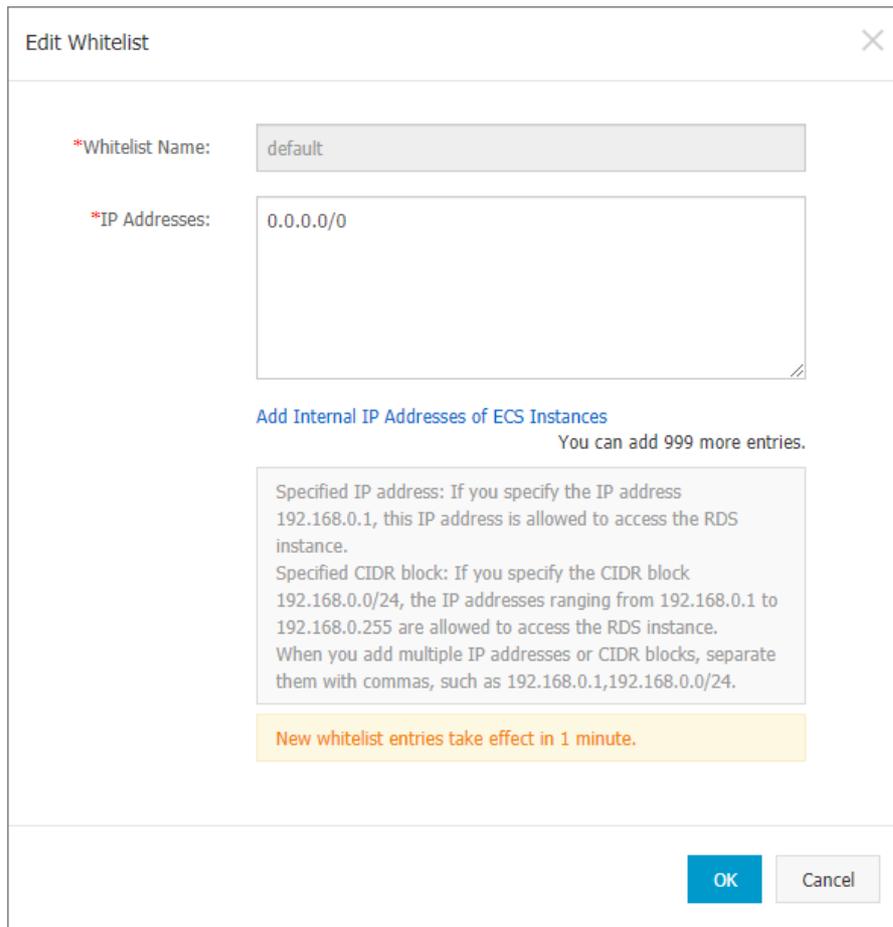
- If you specify the 10.10.10.0/24 CIDR block, IP addresses in the 10.10.10.X format are allowed to access the RDS instance.
- If you want to add multiple IP addresses or CIDR blocks, separate entries with commas (without spaces), such as 192.168.0.1,172.16.213.9.
- After you click **Add Internal IP Addresses of ECS Instances**, the IP addresses of all the ECS instances under your Apsara Stack account are displayed. You can select the required IP addresses to add to the whitelist.

Note If you add a new IP address or CIDR block to the **default** whitelist, the default address 127.0.0.1 is automatically deleted.

Troubleshooting

- Fault description

A stack exception occurs while the system is running, as shown in the following figure.



- Solution
Add the IP address of your region to an RDS whitelist. For more information, see [Specify whitelist settings](#).

3.5. Data development

3.5.1. Create a job

This topic describes how to create a Realtime Compute job.

Procedure

1. [Log on to the Realtime Compute console](#).
2. Click **Development Platform**.
3. Click **Development** in the top navigation bar.
4. Click **Create File** in the toolbar.
5. In the **Create File** dialog box, specify the required fields.

Field	Description
File Name	The name of the file. The specified name must be 3 to 64 characters in length and can contain lowercase letters, digits, and underscores (_). It must start with a lowercase letter.

Field	Description
File Type	The type of the file. Valid values: FLINK_STREAM/SQL and FLINK_STREAM/DATASTREAM.
Storage Path	The folder of the file. You can click the icon on the right side of an existing folder and create a subfolder.

6. Click **OK**.

3.5.2. Development

3.5.2.1. SQL code assistance

The development platform of Realtime Compute offers a complete set of SQL tools in the integrated development environment (IDE). These tools provide the following features to help you with Flink SQL-based development:

- **Syntax check**

On the **Development** page of Realtime Compute, the revised script is automatically saved. When the script is saved, an SQL syntax check is automatically performed. If a syntax error is detected, the **Development** page shows the row and column where the error is located, and the cause of the error.

- **Intelligent code completion**

When you enter SQL statements on the **Development** page of Realtime Compute, auto completion popups about keywords, built-in functions, tables, or fields are automatically displayed.

- **Syntax highlighting**

Flink SQL keywords are highlighted in different colors to differentiate data structures.

3.5.2.2. SQL code version management

Realtime Compute provides key features that help you complete development tasks, such as coding assistance and code version management. A new code version is generated each time you publish a job SQL file. The code version management feature allows you to track code changes and roll back to an earlier version if required.

- **Manage code versions**

On the **Development** page, you can manage Flink SQL code versions. A new code version is generated each time you publish a job SQL file. You can use the code version management feature to track versions, modify the code, and roll the code back to an earlier version.

On the **Versions** tab on the right side of the **Development** page, click **More** in the **Actions** column to manage code versions.

- **Compare**: Check the differences between the current version and an earlier version.
- **Rollback**: Roll back to an earlier version.
- **Delete**: Delete an earlier version.
- **Locked**: Lock the current version.

 **Note** You cannot submit a new version before you unlock the SQL file.

- **Delete code versions**

A snapshot of a code version is created each time you submit an SQL file for publishing a job. This allows you to track code changes. The maximum number of code versions has been specified. If you use Apsara Stack, a maximum of 20 code versions can be published. To find out the maximum number of code versions in other environments, contact the system administrator. If the number of code versions reaches the upper limit, an error message is displayed to alert you to delete one or more earlier versions.

In this scenario, you must delete one or more earlier versions before you publish new versions. To do this, click the **Versions** tab on the right side of the **Development** page, click **More** in the **Actions** column, and select **Delete** to delete expired versions that are no longer needed.

3.5.2.3. Data store management

The **Development** page of the Realtime Compute console provides an easy and effective method to manage data stores. For example, you can register external data stores to reference the data stores.

- **Data preview**

The **Development** page of Realtime Compute allows you to preview data from a wide range of data stores. Data preview allows you to analyze the characteristics of upstream and downstream data, identify key business logic, and complete development tasks with high efficiency.

- **Auto DDL generation**

Realtime Compute provides the auto DDL generation feature. The system can automatically generate DDL statements to reference data stores that can be registered in Realtime Compute. This feature provides a simple method to edit SQL statements for stream processing jobs. This improves overall efficiency and reduces errors when you manually enter SQL statements.

3.5.3. Debug job code

The Realtime Compute development platform provides a simulated running environment where you can customize uploaded data, simulate operations, and check outputs.

After you write the SQL code that implements the computing logic, follow these steps to debug the code:

1. [Log on to the Realtime Compute console.](#)
2. In the top navigation bar, click **Development**.
3. In the left-side navigation pane, click **Development**.
4. On the **Development** tab, double-click the target folder and file name to open the job file.
5. In the top menu bar, click **Syntax Check**.

 **Note** You can use the syntax check feature to check whether the SQL file includes syntax errors. Error messages are displayed for syntax errors.

6. In the top menu bar, click **Debug**. On the **Debug File** page, debug your SQL code.

The test data for debugging can be acquired by using either of the following two methods:

- Upload local data.
 - a. Click **Download Template**.
 - b. Prepare test data based on the template.
 - c. Click **Upload**. After the file is uploaded, you can view the uploaded data in the data preview section.
- Sample online data.
 - a. Click **Random Online Data Sampling** or **Sequential Online Data Sampling**.

- b. View the sampled data in the data preview section.
7. Click **OK**.
8. In the output window, view the debugging result.

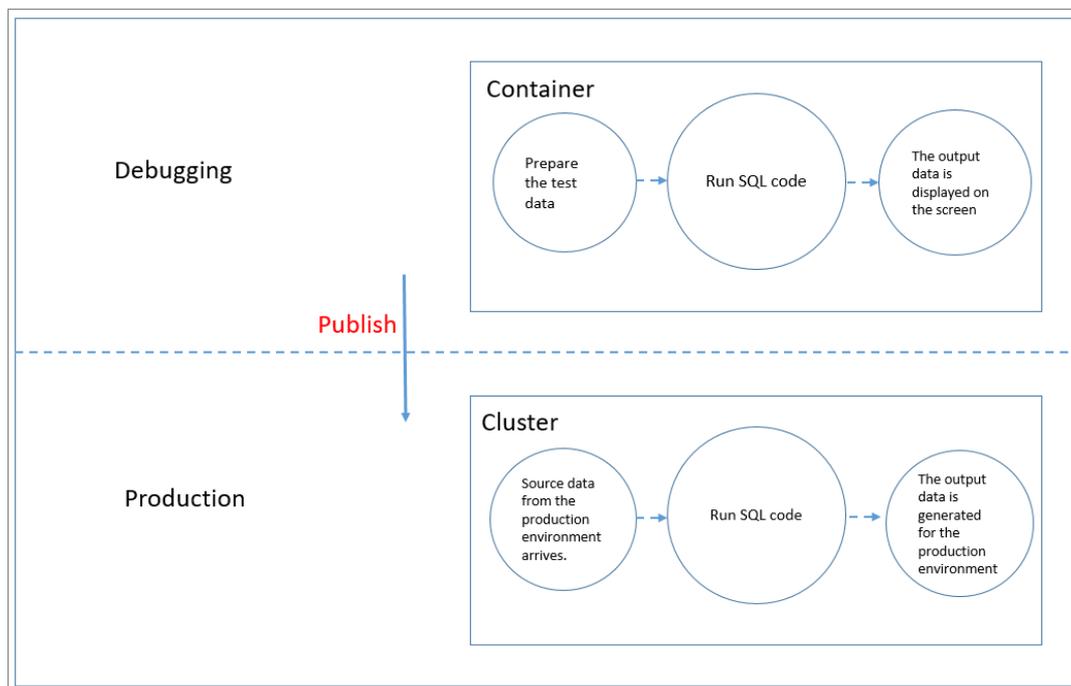
The debugging feature of Realtime Compute provides the following functions:

- Enables isolation between debugging and production environments.

In the debugging environment, the Flink SQL code runs in a separate container, and computing result data is only displayed on the screen of the **Development** page. In this way, the debugging does not affect the running jobs and data stores in the production environment.

In the debugging phase, result data is not written to external data stores. In the production environment, failures may occur due to format errors when result data is written to the target data stores. Such failures cannot be identified or prevented in the debugging phase, and can be detected only while jobs are running. For example, failures may occur in the production environment if your result data is too long. This occurs when the result data is written to a result table in ApsaraDB for RDS and the length of character strings reaches the upper limit for an RDS table. The Realtime Compute team is working on support for writing result data to external data stores in the production environment. This allows you to effectively simulate the production environment and resolve more issues in the debugging phase.

Isolation between debugging and production environments



- Supports the customization of test data.

In the debugging environment, Realtime Compute does not read data from source data stores, such as DataHub topics that store source tables and RDS instances that store dimension tables. You must create a set of test data and upload the test data on the **Development** page.

To make the debugging feature easy to use, Realtime Compute provides a template of test data for each type of job. You can download the template and enter your test data.

Note We recommend that you use the templates to prevent errors.

- Specifies a separator.

A comma (,) is used as the separator in files for debugging by default. An example of a file for debugging is described as follows:

```
id,name,age
1,alicloud,13
2,stream,1
```

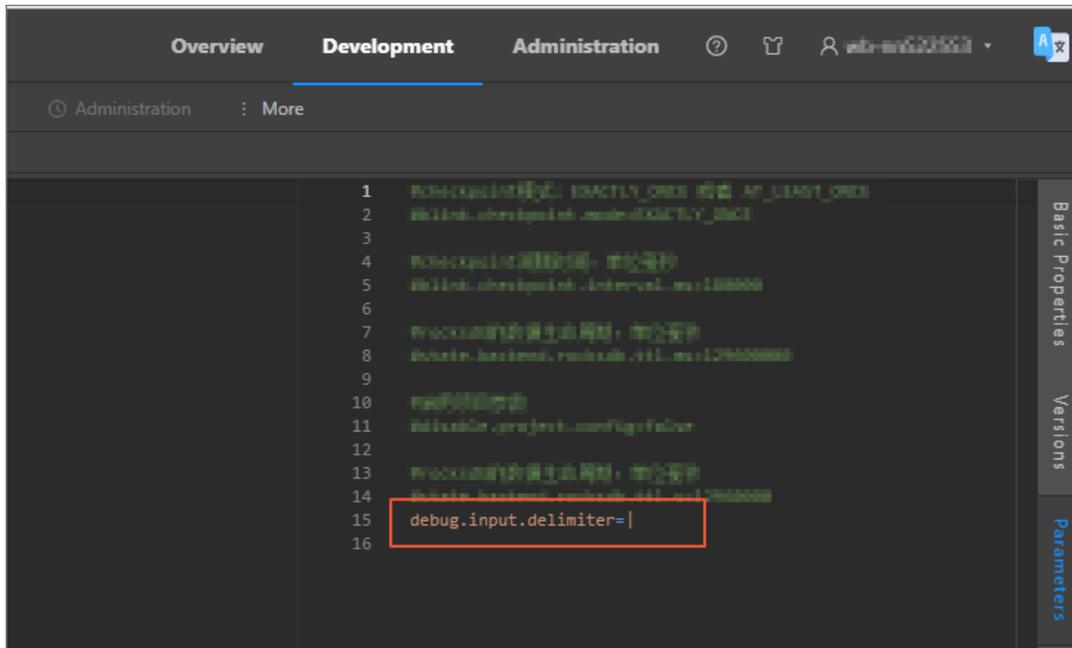
If the separator is not specified, a comma (,) is used to separate fields. If you need to use a JSON string as the field data and the string contains commas (,), you must specify another character as the separator.

Note Realtime Compute allows you to specify a character as the separator, but not a multi-character string, such as aaa.

```
id|name|age
1|alicloud|13
2|stream|1
```

In this example, specify the debug.input.delimiter parameter as follows: `debug.input.delimiter=|`.

Specify a separator



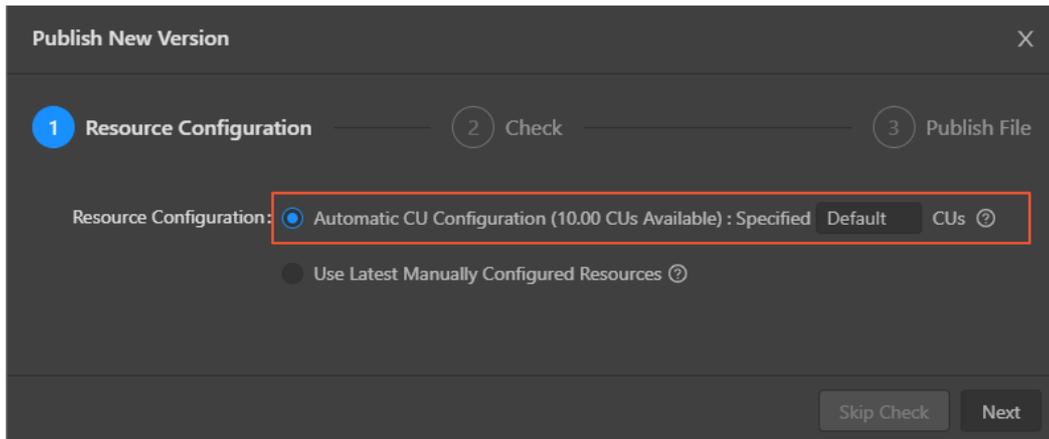
3.5.4. Publish a job SQL file

After you have created and debugged a job Flink SQL file, you can publish the SQL file and manage the job in the production environment.

Procedure

1. [Log on to the Realtime Compute console.](#)
2. In the top navigation bar, Click **Development**.
3. In the top menu bar, click **Publish**.
4. In the dialog box that appears, select **Automatic CU Configuration**. If you are performing automatic configuration for the first time, we recommend that you use the default number of CUs. Click **Next**.

Configure resources



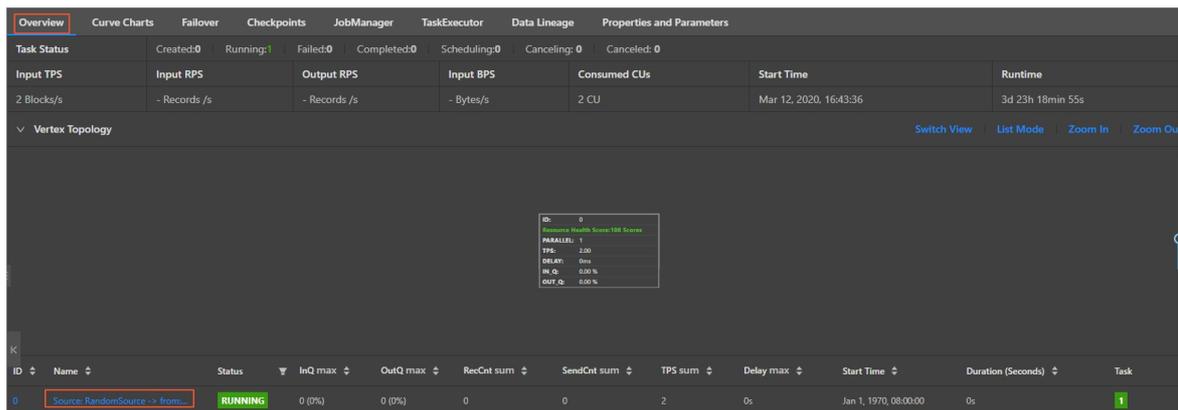
5. Check the data. After the check is completed, click **Next**.
6. Click **Publish**.
7. Go to the **Administration** page to start the job.
 - i. In the top navigation bar, click **Administration**.
 - ii. On the **Administration** page, find the target job, and click **Start** in the **Actions** column.

3.5.5. View logs

You can view job logs to check the job run information. This topic describes how to view job logs.

Procedure

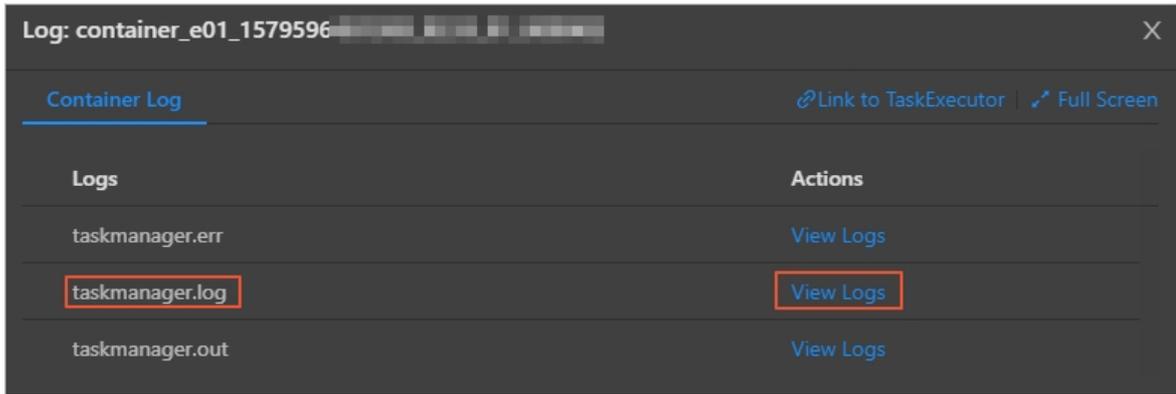
1. Go to the **Administration** page in Realtime Compute.
 - i. [Log on to the Realtime Compute console](#).
 - ii. In the top navigation bar, click **Administration**.
 - iii. On the **Jobs** page that appears, click the target job name in the **Job Name** column.
2. At the bottom of the **Overview** page, click the name of the target vertex.



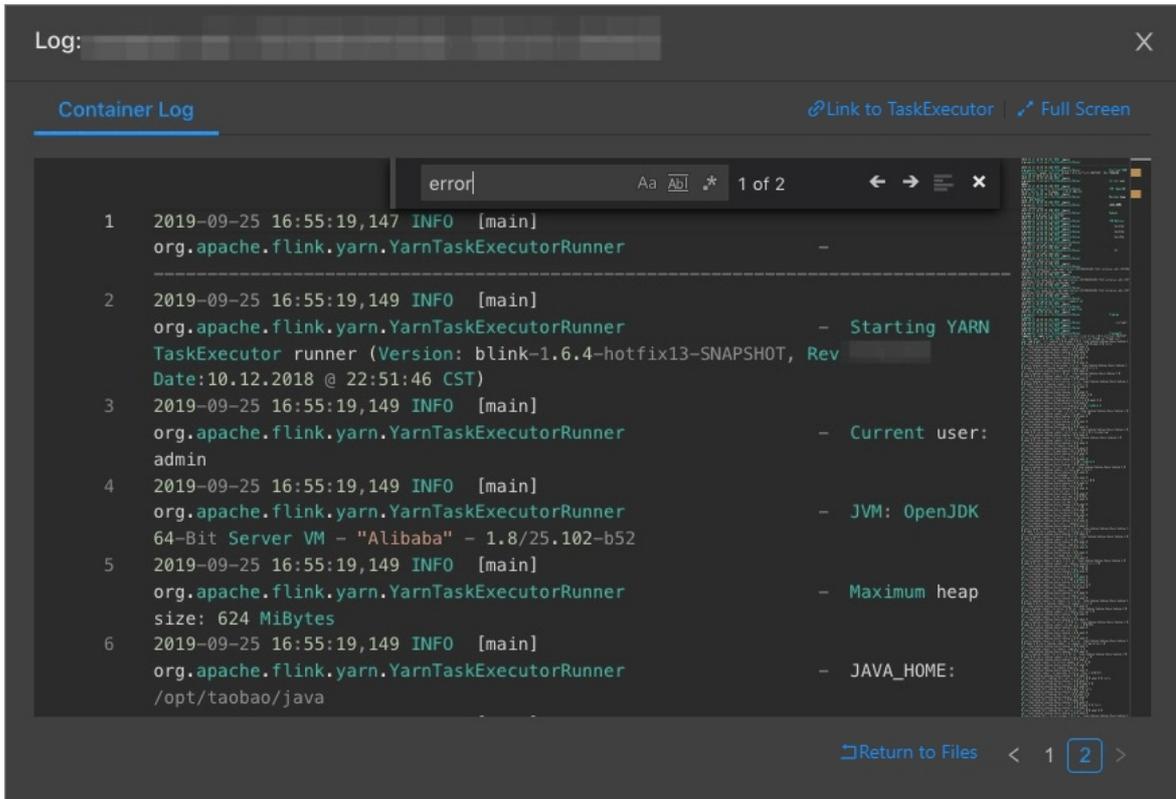
3. On the **Execution Vertex** page, click the **Subtask List** tab, and click **View Logs** in the **Actions** column.



4. In the **Log** dialog box that appears, click **View Logs** for **taskmanager.log** in the **Actions** column.



5. On the Container Log tab, view the log entries.



Note You can press Ctrl+F for Windows or Cmd+F for MacOS to search for specified log entries. We recommend that you view the logs from the last page. The first **error** log entry describes the root cause of the job error.

4. Machine Learning Platform for AI

4.1. What is machine learning?

Machine learning is a process of using statistical algorithms to learn large amounts of historical data and generate an empirical model to provide business strategies.

Apsara Stack Machine Learning Platform for AI is a set of data mining, modeling, and prediction tools. It is developed based on MaxCompute (also known as ODPS). Machine Learning Platform for AI supports the following functions:

- Provides an all-in-one algorithm service covering algorithm development, sharing, model training, deployment, and monitoring.
- Allows you to complete the entire procedure of an experiment either through the GUI or by running PAI commands. This function is typically intended for data mining personnel, analysts, algorithm developers, and data explorers.
- In Apsara Stack, Machine Learning Platform for AI runs on MaxCompute. Machine Learning Platform for AI allows you to call algorithms to decouple the applications and compute engines after you have deployed algorithm packages in MaxCompute clusters.
- Provides various algorithms and reliable technical support, providing more options to resolve service issues. In the Data Technology (DT) era, you can use Machine Learning Platform for AI to implement data-driven services.

Machine Learning Platform for AI can be applied in the following scenarios:

- Marketing: commodity recommendations, user profiling, and precise advertising.
- Finance: loan delivery prediction, financial risk control, stock trend prediction, and gold price prediction.
- Social network sites (SNSs): microblog leader analysis and social relationship chain analysis.
- Text: news classification, keyword extraction, text summarization, and text analysis.
- Unstructured data processing: image classification and image text extraction through OCR.
- Other prediction cases: rainfall forecast and football match result prediction.

Machine learning can be divided into three types:

- Supervised learning: Each sample has an expected value. You can create a model and map input feature vectors to target values. Typical examples of this learning mode include regression and classification.
- Unsupervised learning: No samples have a target value. This learning mode is used to discover potential regular patterns from data. Typical examples of this learning mode include simple clustering.
- Reinforcement learning: This learning mode is complex. A system constantly interacts with the external environment to obtain external feedback and determines its own behavior to achieve a long-term optimization of targets. Typical examples of this learning mode include AlphaGo and driverless vehicles.

4.2. Quick start

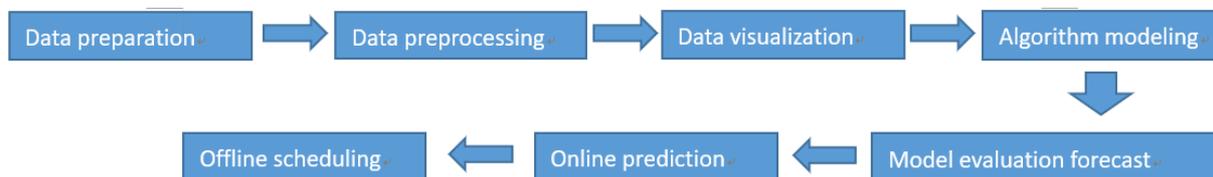
4.2.1. Overview

This topic describes how to perform data preparation, data preprocessing, data visualization, algorithm modeling, model prediction and evaluation, online prediction, and DataWorks task scheduling to set up a machine learning experiment.

Note This document covers Apsara Stack Machine Learning Platform for AI, online model service, and deep learning framework. The online model service and deep learning framework are not basic functions of Apsara Stack Machine Learning Platform for AI and must be purchased separately.

For more information, see [Machine learning experiment creation flowchart](#).

Machine learning experiment creation flowchart



1. Data preparation

Import target data into the Apsara Stack Machine Learning Platform for AI console.

2. Data preprocessing

Perform data processing, including SQL-based conversion, normalization, and standardization, to ensure that all data has the same dimensions.

3. Data visualization

Display data in charts to view the features of the data and the distribution of the values. This serves as the basis for model algorithm selection.

4. Algorithm modeling

Use machine learning algorithms to train data and ultimately build a model.

5. Model evaluation forecast

Make predictions from and evaluate the model, and use the prediction results to create business development strategies.

6. Online prediction

Use online prediction to deploy the generated model and adjust your business strategy based on the prediction results.

7. Offline scheduling

Deploy experiments in DataStudio and run them on a regular basis.

4.2.2. Log on to the Machine Learning Platform for AI console

This topic describes how to log on to the Machine Learning Platform for AI console.

Prerequisites

- The IP address or domain name of the Apsara Uni-manager Management Console is obtained from deployment engineers before you log on to the Apsara Uni-manager Management Console. The URL for logging on to the Apsara Uni-manager Management Console is in the following format: `http://IP address or domain name of the Apsara Uni-manager Management Console/manage`.
- A browser is available. We recommend that you use Google Chrome.

Procedure

1. Enter the URL of the Apsara Uni-manager Management Console in the address bar and press the Enter key.
2. Enter your username and password.

The system has a default super administrator, whose username is super. The super administrator can create system administrators. A system administrator can create system users and notify the users of the default password by SMS or email.

Note When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username as prompted. For security concerns, your password must meet the minimum complexity requirements. The password must be 8 to 20 characters in length and must contain at least two types of the following characters: letters, digits, and special characters such as exclamation points (!), at signs (@), number signs (#), dollar signs (\$), and percent signs (%).

3. Click **Login** to go to the homepage of the Apsara Uni-manager Management Console.
4. On the top navigation bar, choose **Big Data > Machine Learning**.
5. On the page that appears, set the **Department** and **Region** parameters and click **PAI** to go to the Apsara Stack Machine Learning Platform for AI console.

Note If this is the first time that you log on to the Apsara Stack Machine Learning Platform for AI console, you must perform the following steps:

- i. Add an organization
Create an organization to store resource sets and their resources.
- ii. Create a user
Create users and assign the users different roles to meet different requirements for system access control.
- iii. Create a resource set
Create a resource set before you apply for resources.
- iv. Add a member to a resource set
Add users to the resource set.
- v. Go to the homepage of the Apsara Uni-manager Management Console. In the left-side navigation pane, choose **Big Data > MaxCompute**. Then, create a task account and a project in MaxCompute.
 - a. Create an account: Set the **Department** parameter to the department you created in Step 1.
 - b. Create a MaxCompute project: Set the **Department** parameter to the department you created in Step 1. Set the **Dt_project** parameter to the project you created in Step 2. Set the **MaxCompute Task Account** parameter to the task account that you created.
- vi. Create a Dataworks workspace. In the **Advanced Settings** section, set the **MaxCompute Project Name** parameter to the name of the project that you created in the previous step.

4.2.3. Data preparation

This topic describes how to import data into the Apsara Stack Machine Learning Platform for AI console for modelling.

Prerequisites

Make sure that you have created a MaxCompute project and imported table data into the project. You can

download the data from [Download Data](#).

Procedure

1. [Log on to the Apsara Stack Machine Learning Platform for AI console](#) and click **Experiments** in the left-side navigation pane.
2. On the **Experiments** page, right-click **My Experiments** and choose **New Experiment** from the short cut menu. In the dialog box that appears, enter the experiment name and description. Click **Create** to go to the **Components** page.
3. In the **Components** list, click **Data Source/Target**, and drag and drop the **Read MaxCompute Table** onto the canvas.
4. Click the **Read MaxCompute Table** component and configure its parameters. Enter the MaxCompute table name in **Table Name** in the right-side configuration pane.
5. In the right-side parameter setting pane, click **Column Information** to view the column name, data type, and the first 100 rows of data in the input table.

4.2.4. Data preprocessing

This topic describes how to perform data preprocessing by using methods such as normalization, SQL scripts, and data splitting.

Prerequisites

Before data preprocessing, make sure that you have completed [data preparation](#).

Procedure

1. [Log on to the Apsara Stack Machine Learning Platform for AI console](#) and click **Components** in the left-side navigation pane.
2. In the **Components** list, click **Tools**. Drag and drop the **SQL Script** component onto the canvas. Click **Data Preprocessing**. Drag and drop the **Normalization** component onto the canvas, and connect the components.
3. Click the **SQL Script** component and click the right-side **Parameters** tab. In the **SQL Script** input box, enter the following SQL scripts to convert the features from string type to numeric type.

```
select age,
(case sex when 'male' then 1 else 0 end) as sex,
(case cp when 'angina' then 0 when 'notang' then 1 else 2 end) as cp,
trestbps,
chol,
(case fbs when 'true' then 1 else 0 end) as fbs,
(case restecg when 'norm' then 0 when 'abn' then 1 else 2 end) as restecg,
thalach,
(case exang when 'true' then 1 else 0 end) as exang,
oldpeak,
(case slop when 'up' then 0 when 'flat' then 1 else 2 end) as slop,
ca,
(case thal when 'norm' then 0 when 'fix' then 1 else 2 end) as thal,
(case status when 'sick' then 1 else 0 end) as ifHealth
from ${t2};
```

4. Click the **Normalization** component and select all fields to normalize the numeric features to values ranging from 0 to 1.
5. Click **Data Preprocessing**. Drag and drop the **Split** component onto the canvas and set **Split Ratio** to **0.7**.

 **Note** This step splits data into two parts: 70% of the data is used as the model training set, and 30% of the data is used as the model prediction set.

4.2.5. Data visualization

This topic describes how to view the features and value distribution by using statistical analysis components.

Prerequisites

Before data visualization, make sure that you have completed [data preprocessing](#).

Procedure

1. [Log on to the Apsara Stack Machine Learning Platform for AI console](#) and click **Components** in the left-side navigation pane.
2. In the **Components** list, click **Statistical Analysis**. Drag and drop the **Whole Table Statistics** component onto the canvas. Connect the components, and click **Run** at the bottom of the canvas.
3. After the experiment stops running, right-click **Whole Table Statistics** and choose **View Data** from the short cut menu. The analysis report is displayed.

4.2.6. Algorithm modeling

This topic describes how to perform feature training and generate models by using the machine learning components.

Prerequisites

Before algorithm modeling, ensure that you have completed [data preprocessing](#) and learned the data characteristics and value distribution through [data visualization](#).

Procedure

1. [Log on to the Apsara Stack Machine Learning Platform for AI console](#) and click **Components** in the left-side navigation pane.
2. In the **Components** list, choose **Machine Learning > Binary Classification**. Drag and drop the **Binary Logistic Regression** component onto the canvas, and connect the corresponding components and data streams.
3. Click the component, and select 13 feature columns from **Training Feature Columns** in the right-side **Column Settings** pane. All parameters use the default settings.
4. Click **Run**.
5. Click **Models** in the left-side navigation pane to view the generated model.

4.2.7. Model prediction evaluation

This topic describes how to use a model to make predictions and evaluate its results by using the prediction and evaluation components.

Prerequisites

Before evaluating the prediction, make sure that you have completed [algorithm modeling](#) and generated a machine learning model from the experiment.

Procedure

1. Log on to the [Apsara Stack Machine Learning Platform for AI console](#) and click **Components** in the left-side navigation pane.
2. In the **Components** list, click **Machine Learning**. Drag and drop the **Prediction** component onto the canvas, and connect the corresponding components and data streams.
3. Choose **Machine Learning > Evaluation**. Drag and drop the **Binary Classification Evaluation** component onto the canvas and connect the corresponding components and data streams.
4. Click **Run** in the upper-left corner of the canvas. During the running process, select a component and click the **Developer Tool** () icon in the lower-right corner of the canvas to view the running status of the component.
5. Right-click the **Binary Classification Evaluation** component and choose **View Evaluation Report** from the shortcut menu to generate the ROC curve of the LR model trained with different parameters.

4.2.8. DataWorks task scheduling

After you have run all nodes in an experiment, you can deploy the experiment to DataWorks and schedule DataWorks to periodically run the experiment. This topic uses air quality prediction as an example scenario.

Prerequisites

Before scheduling an experiment, you must make sure that you have successfully run all nodes in the experiment and that the experiment is deployed to DataWorks.

Procedure

1. Log on to the [Apsara Stack Machine Learning Platform for AI console](#). Click **Experiments** in the left-side navigation pane.
2. Click the **My Experiments** tab and select an experiment to navigate to the canvas.

 **Notice** Make sure all components have been run in the experiment. A green check means that the component is running correctly.

3. In the upper-left corner of the canvas, choose **Deploy > Schedule DataWorks Tasks** to go to DataStudio.
4. In the DataStudio console, choose **Create > Algorithm > Machine Learning Platform for AI**, and then create a Machine Learning experiment node.
5. In the **Create Node** dialog box, enter the node name, select the target folder, and click **Submit**.

 **Notice** You must select a target folder for the algorithm type.

After the experiment node is created.

6. Select the experiment from the drop-down list.
7. Configure task scheduling parameters, including the recurrence, input, and output parameters.
8. Click **Submit**. The task will be executed the next day.
9. Click **Administration** in the upper-right corner to go to the administration page. You can view the status of the machine learning task and the system log. You can also perform other operations such as adding retroactive data and testing the experiment.

4.3. Online model service (must be activated separately)

4.3.1. Deploy an online model service

This topic describes how to deploy the generated experiment model through the online model service. You can adjust your business strategy anytime based on predicted results.

Prerequisites

Before deploying the online model service, make sure that the preceding steps are completed and the components are running properly. A green check means that the component is running correctly.

Procedure

1. Log on to the [Apsara Stack Machine Learning for AI console](#) and click **Experiments** in the left-side navigation pane.
2. Click the **My Experiments** tab and select an experiment to navigate to the canvas.

 **Notice** Make sure that the selected experiment is running properly. A green check means that the component is running correctly.

3. In the upper-left corner of the canvas, choose **Deploy > Online Model Service**.
4. Select the model to deploy and click **Next**.
5. Select a deployment mode. You can select one of the following modes:
 - o [New Service](#)
 - o [Add Existing Service Version](#)
 - o [Create Blue-green Deployment](#)

4.3.2. Create a service

This topic describes how to use the **New Service** mode to deploy online prediction services.

Procedure

1. Complete [Preparations for online model prediction](#).
2. Set **Processes** and **Quota**.

 **Note** **Processes** determines the maximum number of concurrently running programs. **Quota** determines the running speed and the parameters such as RT and QPS.

3. Click **Deploy**. It takes several minutes to create the model.
4. After the model is created, click the model name to view information about the model invocation.
5. Click the icons under **Monitor** to view statistics about QPS, response, RT, traffic, CPU utilization, memory usage, and daily invocation.

6.  **Note** Perform this step when resources are insufficient and need to be expanded.

Click **Update** to expand resources.

7. Click **Online Debugging** in the upper-right corner of the page and select the current model.

4.3.3. Add an existing service version

This topic describes how to use the **Add Existing Service Version** mode to deploy online prediction services.

Prerequisites

Before you use the **Add Existing Service Version** mode to deploy online prediction services, ensure that you have deployed one version of online prediction services through the [New Service](#) mode.

Procedure

1. Complete [Preparations for online model prediction](#).
2. Select **Add Existing Service Version**.
3. Select a deployed model. It takes several minutes to add a version.
4. After the model is deployed, select the added version from the **Current Version** drop-down list.

4.3.4. Create a blue-green deployment

This topic describes how to use the **Create Blue-green Deployment** mode to deploy online prediction services.

Prerequisites

Before you use the **Create Blue-green Deployment** mode to deploy online prediction services, ensure that you have deployed two versions of online prediction services through the [New Service](#) and [Add Existing Service Version](#) modes.

Context

In the blue-green deployment mode, you can deploy and test the target version without stopping the source version. After confirming that the target version is running normally, switch all traffic to the target version. Blue-green deployment is safe and does not interrupt services.

Procedure

1. Complete [Preparations for online model prediction](#).
2. Select the deployed model service and click **Deploy**. The deployment may take several minutes.
3. Click **Switch Traffic** and adjust the ratio of traffic forwarded to the two models. The initial ratio is 100% for both models.
4. Perform online debugging.
 - i. Click **Online Debugging** in the upper-right corner of the page and select the deployed model.
 - ii. Enter data (feature input) in **Body**. For example, the body information of the logistic regression model for heart disease prediction is as follows:

```
[{"sex":0,"cp":0,"fbs":0,"restecg":0,"exang":0,"slope":0,"thal":0,"age":0,"trestbps":0,"chol":0,"thalach":0,"oldpeak":0,"ca":0}]
```

- iii. Click **Run** and check the result.

4.4. Components

4.4.1. Overview

This topic describes how to use and configure machine learning components. When building a machine learning experiment, you can select components based on the features of existing data to generate a model and make accurate predictions for your business.

Each component has one or more input or output ports. You can move the pointer over the ports to view their descriptions and connect the components.

4.4.2. Data source and target

This topic describes components in the **Data Source/Target** category, such as the Read MaxCompute Table and Write MaxCompute Table components.

Read MaxCompute tables

You can use the Read MaxCompute Table component to read MaxCompute tables. By default, this component reads data of the current project. If you want to read data from tables in another project for which you have access, you can prefix the table name with the project name in the `project name.table name` format. For example, `tianchi_project.weibo_data`. After you specify the input table, the system reads the structural data of the table. You can click the **Column Information** tab to view the data. This component does not support views.

If the selected input table is a partitioned table, the back end automatically selects the Partition checkbox. You can select or configure partition parameters. Only one partition can be selected. If you do not select the Partition checkbox or do not specify the partition parameters, the whole table is selected. If the input table is non-partitioned, the Partition checkbox cannot be selected.

Write MaxCompute tables

You can use the Write MaxCompute Table component to write data to tables in the current project or tables in other projects. This component can write data to partitions. Partitions must be created for the table in the MaxCompute console before this component can write data to the partitions. You can set the table lifecycle measured in days.

4.4.3. Data preprocessing

4.4.3.1. Sampling and filtering

4.4.3.1.1. Random sampling

Data is sampled randomly and independently. You can specify a ratio or quantity of samples to be taken and choose whether to enable sampling with replacement.

Parameter settings

Parameters Setting
Tuning

Sample Size Specify either the sample size or...

Sampling Fraction Range: (0,2). Specify either...

Sampling with Replacement

Random Seed Positive integer.

PAI command

```

Pai -name sample
    -project algo_public
    -DinputTableName=wbpc
    -DoutputTableName=wbpc_sample
    -Dratio=0.3;
  
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions in the input table are selected by default.
ratio	Required. The sampling fraction.	(0, 1)	-
outputTableName	Required. The name of the output table.	-	-
outputTablePartition	Optional. The partition of the output table.	-	The output table is a non-partitioned table by default.

Parameter	Description	Valid values	Default value
lifecycle	Optional. The lifecycle of the output table.	A positive integer in the range of [1, 3650]	No lifecycle is set by default.

4.4.3.1.2. Weighted sampling

Sample data is collected based on weights. The weight column must be of double or int type. Data is sampled based on the value of its corresponding weight. For example, data with a col value of 1.2 has a higher probability to be sampled than data with a col value of 1.0.

Parameter settings

Parameters Setting

Sample Size Specify either the sample size or...

Sampling Fraction Range: (0,1). Specify either...

Sampling with Replacement

Weight Columns Double or bigint type.

Random Seed Positive integer.

Parameter settings

Parameter	Description
Sample Size	You can specify the number of samples to be taken, which is 10,000 by default. For sampling without replacement, the number of samples cannot be greater than the number of data entries.
Sampling Fraction	You can use either the Sample Size or Sampling Fraction parameter. You can choose sampling with or without replacement, the latter of which is used by default. Select the checkbox to enable sampling with replacement.
Weight Columns	You can select a weight column from the drop-down list. The weight column can be of the double or bigint type.
Random Seed	The random seed, which is a positive integer. This parameter is empty by default.

- You can choose sampling with or without replacement, the latter of which is used by default. Select the checkbox to enable sampling with replacement.
- You can specify the number of samples to be taken, which is 10,000 by default.

Note For sampling without replacement, the number of samples cannot be greater than the number of data entries.

- You can select a weight column from the drop-down list. The weight column can be of the double or bigint type.

PAI command

```
PAI -name WeightedSample
-project algo_public
-DprobCol="previous"
-DsampleSize="500"
-DoutputTableName="test2"
-DinputPartitions="pt=20150501"
-DinputTableName="bank_data_partition";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
replace	Indicates whether sampled data is replaced. If this parameter is set to true, data is replaced after it is sampled. If this parameter is set to false, data is not replaced after it is sampled.
probCol	The columns to be weighted. Each value indicates the weight of an entry. Normalization is not required.
sampleSize	The number of samples to be taken. For sampling without replacement, the number of samples cannot be greater than the number of data entries.
outputTableNames	The name of the output table. Separate multiple table names with commas (,).
inputPartitions	Optional. The partitions selected from the input table for training. If no partitions are specified, the entire table is selected.
inputTableName	The name of the input table.
replace	Optional. This parameter indicates whether sampled data is replaced. If this parameter is set to true, data is replaced after it is sampled. If this parameter is set to false, data is not replaced after it is sampled.

4.4.3.1.3. Filtering and mapping

You can filter data based on filtering expressions and rename columns.

Parameter settings

- Use the WHERE condition to filter data similar to how it would function in an SQL statement.

Filtering conditions: Operators available include the equal (=), not equal (!=), greater than (>), less than (<), greater than or equal to (>=), less than or equal to (<=) signs, as well as like and rlike.

- Rename columns.

PAI command

```
PAI -name Filter
-project algo_public
-DoutTableName="test_9"
-DinputPartitions="pt=20150501"
-DinputTableName="bank_data_partition"
-Dfilter="age>=40";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
outTableName	The name of the output table.
inputPartitions	Optional. The partitions selected from the input table for training. If no partitions are specified, the entire table is selected.
inputTableName	The name of the input table.
filter	The WHERE condition to filter data. Operators available include the equal (=), not equal (!=), greater than (>), less than (<), greater than or equal to (>=), less than or equal to (<=) signs, as well as like and rlike.

4.4.3.1.4. Stratified sampling

Stratified sampling is a statistical computing method. It works by dividing a population into several strata based on specified features, performing random sampling at each stratum, and creating a sample collection.

Parameter settings

Parameter	Description
Column Settings	Stratification Column: Required. Samples are stratified based on this column.
Parameter Settings	Sampling Fraction/Sample Size: Required. A value less than 1 represents the sampling fraction per stratum. A value greater than 1 represents the number of samples at each stratum.
	Other Sampling Configurations: Optional. This parameter allows you to collect different numbers of samples at different strata.
	Random Seed: Optional. Valid values: 1, 2, 3, 4, 5, 6, and 7.

PAI command

```
Pai -name sample
-project algo_public
-DinputTableName=wbpc
-DoutputTableName=wbpc_sample
-DstrataColName="label"
-DsampleSize="A:200,B:300,C:500"
-DrandomSeed=1007
-Dlifecycle=30
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	-	-
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions in the input table are selected by default.
<code>strataColName</code>	Required. The stratification column.	-	-
<code>outputTableName</code>	Required. The name of the output table.	-	-
<code>sampleSize</code>	Optional. An integer value that specifies the number of samples taken from each stratum. A string value must be in the <code>strata0:n0,strata1:n1...</code> format. Each item in the string represents the number of samples to be taken from the corresponding stratum.	-	

Parameter	Description	Valid values	Default value
<code>sampleRatio</code>	Optional. A decimal value from 0 to 1 that represents the ratio of data for each stratum to be sampled. A string value must be in the <code>strata0:r0,strata1:r1...</code> format. Each item in the string represents the sampling fraction for the corresponding stratum.	-	-
<code>randomSeed</code>	Optional. The number of random seeds.	-	0
<code>lifecycle</code>	Optional. The lifecycle of the output table.	A positive integer in the range of [1, 3650]	No lifecycle is set by default.
<code>coreNum</code>	Optional. The number of cores.	-	Automatically calculated.
<code>memSizePerCore</code>	Optional. The memory size of each core.	-	Automatically calculated.

4.4.3.2. Data merge

4.4.3.2.1. Join

This component merges two tables by associating the information in the tables and outputting the specified columns. This component is similar to the JOIN statement of SQL.

Parameter settings

- Join types: left join, internal join, right join, and full join.
- Only the equation condition is supported.
- You can manually add or delete join conditions.

PAI command

No PAI command is available.

4.4.3.2.2. Merge columns

You can merge data of two tables by column. The two tables must have the same number of rows.

Parameter settings

Procedure

1. Select input columns from the left table.
2. Select input columns from the right table.

When merging columns:

- The two tables must have the same number of rows.
- The names of output columns selected from the left and right tables cannot be the same.
- When selecting an output column, you can change its name.
- If no output columns are selected from the left or right table, the whole table is selected. In this case, if **Automatically Rename Output Columns** is selected, the duplicate columns are renamed and then output.

PAI command

```
PAI -name AppendColumns
-project algo_public
-DoutputTableColNames="petal_length,petal_width,petal_length2,petal_width2"
-DautoRenameCol="false"
-DoutputTableName="pai_temp_770_6840_1"
-DinputTableNames="iris_twopartition,iris_twopartition"
-DinputPartitionsInfoList="dt=20150125/dp=20150124;dt=20150124/dp=20150123"
-DselectedColNamesList="petal_length,petal_width;sepal_length,sepal_width";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
outputTableColNames	The names of the columns in the new table. The column names must be separated with commas (.). If autoRenameCol is set to true, this parameter is ignored.
autoRenameCol	Optional. This parameter specifies whether to automatically rename the columns in the output table. If the value is true, the columns are renamed. If the value is false, the columns are not renamed. Default value: false.
outputTableName	The name of the output table.
inputTableNames	The name of the input table. Separate multiple table names with commas (.).
inputPartitionsInfoList	Optional. A list of partitions selected from the corresponding input tables. Partitions of the same table must be separated with commas (,) and partitions of different tables must be separated with semicolons (;).
selectedColNamesList	The names of selected columns. The names of columns in the same table must be separated with commas (,) and the names of columns in different tables must be separated with semicolons (;).

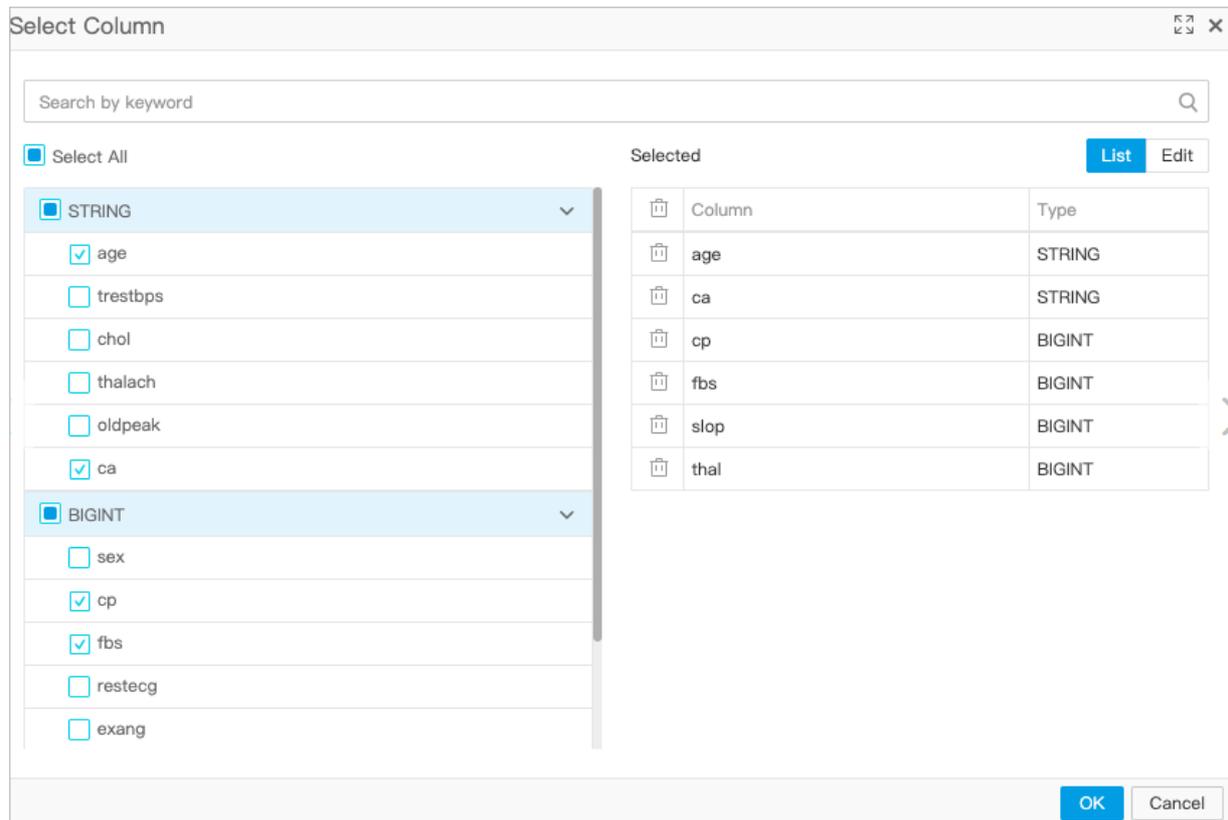
4.4.3.2.3. Merge rows (UNION)

To merge the data of two tables by row, the quantity and data type of the output columns selected from the left and right tables must be the same. The function is integrated with the UNION and UNION ALL functions.

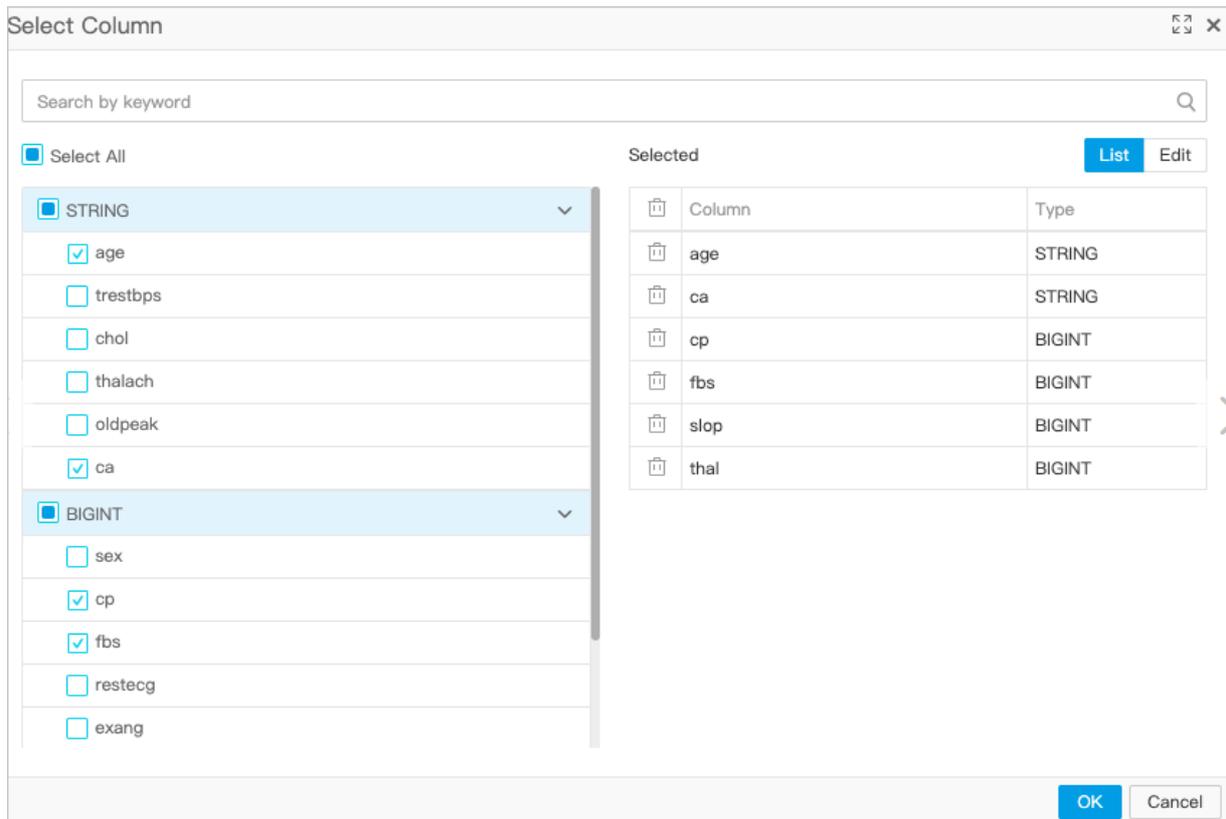
Parameter settings

- During the merge process, the numbers of columns selected from the left and right tables must be the same, and the data types of the corresponding columns must be the same.
- You can enter conditions in the text box by which to filter and select columns. The whole table is selected by default. Operators available include the equal (=), not equal (!=), greater than (>), less than (<), greater than or equal to (>=), and less than or equal to (<=) signs, as well as like and rlike.
- **Remove Duplicates** is selected by default. When this option is selected, duplicate rows in the output table are removed.

The following figure shows the union columns selected from the left table.



The following figure shows the union columns selected from the right table.



PAI command

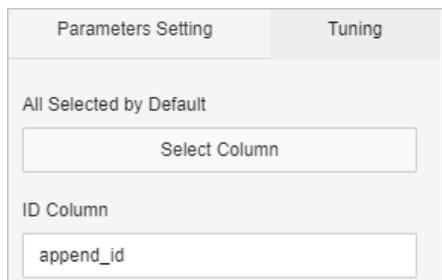
No PAI command is available.

4.4.3.3. Others

4.4.3.3.1. Add ID column

You can append an ID column to a table as the first column and save the table as a new table.

Parameter settings



PAI command

```
PAI -name AppendId
-project algo_public
-DIDColName="append_id"
-DoutputTableName="test_11"
-DinputTableName="bank_data"
-DselectedColNames="age,campaign,cons_conf_idx,cons_price_idx,emp_var_rate,euribor3m,nr_employed,pdays,poutcome,previous,y";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
IDColName	The name of the appended ID column. ID numbers start from 0 and increment by one. Example: 0, 1, 2, 3, ...
outputTableName	The name of the output table.
inputTableName	The name of the input table.
selectedColNames	The names of the columns to be retained. Separate multiple columns with commas (,).

4.4.3.3.2. Split

This component is used to split an input table or a partition based on a specified ratio, and output two tables from two output ports.

Algorithm component

Parameter settings

- The Split component has two output ports.
- In Parameter settings, if the splitting fraction is set to 0.7, the left output port outputs 70% of the data and the right output port outputs 30% of the data.

PAI command

```
pai -name split -project algo_public
-DinputTableName=wbpc
-Doutput1TableName=wbpc_split1
-Doutput2TableName=wbpc_split2
-Dfraction=0.25;
```

Parameter settings

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-

Parameter	Description	Valid values	Default value
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions in the input table are selected by default.
output1TableName	Required. The name of output table 1.	-	-
output1TablePartition	Optional. The partitions in output table 1.	-	Output table 1 is a non-partitioned table by default.
output2TableName	Required. The name of output table 2.	-	-
output2TablePartition	Optional. The partitions in output table 2.	-	Output table 2 is a non-partitioned table by default.
fraction	Required. The portion of data diverted to output table 1.	(0, 1)	-
lifecycle	Optional. The lifecycle of the output table.	A positive integer in the range of [1, 3650]	No lifecycle is set by default.

4.4.3.3.3. Missing value imputation

This component replaces a null value or a specified value with the maximum, minimum, average, or custom value. A list of values is defined to impute the missing values in an input table with the specified values.

- This component can replace a numeric null value with the maximum, minimum, average, or custom value.
- This component can also replace a null string, empty string, null and empty string, or specified value with a custom value.
- The missing values to be imputed can be null strings, empty strings, or custom values. If you choose empty strings, the data type of the target column must be string.

The parameters for the two input ports are as follows:

- **inputTableName**: the name of the input table for which to replace missing data.
- **inputParaTableName**: the name of the input configuration table that contains parameters generated by the missing value imputation node. Based on this parameter, configuration parameters in one table can be applied to a new table.

Parameters for the two output ports are as follows:

- **outputTableName**: the name of the imputed output table.

- **outputParaTableName:** the name of the output configuration table, which can be applied to other datasets.
- **Columns to Impute:** the names of the columns for which to replace missing values.
- **Original Value:** the values to be replaced.
- **Replaced With:** the replacement values.

PAI command

```
PAI -name FillMissingValues
  -project algo_public
  -Dconfigs="poutcome,null-empty,testing" \
  -DoutputTableName="test_3"
  -DinputPartitions="pt=20150501"
  -DinputTableName="bank_data_partition";
```

Algorithm parameters

Parameter	Description	Valid value	Default value
inputTableName	Required. The name of the input table.	Table name	N/A
inputTablePartitions	Optional. The partitions selected from the input table for training.	Partition name	The whole table is selected by default.
outputTableName	Required. The name of the output table.	Table name	N/A
configs	Required. The configurations for missing value imputation. Example: <code>col1. null. 3.14; col2. emotv. hello: col3, empty-null, world</code> , where <i>null</i> indicates a null value and <i>empty</i> indicates an empty string. If you choose to use empty strings to fill the target columns, the data type of the target column must be string. The variables used to specify the replacement value as maximum, minimum, and average are <code>max</code> , <code>min</code> , and <code>mean</code> respectively. If you want to impute a custom value to the target column, use a user-defined variable in the <code>col4.user-defined,str,str123</code> format.	N/A	N/A

Parameter	Description	Valid value	Default value
outputParaTableName	Required. The name of the output configuration table.	Table name	N/A
inputParaTableName	Optional. The name of the input configuration table.	Table name	No input configuration table is set by default.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	A positive integer	Automatically calculated.

Examples

Test data

- SQL statement to generate data:

```
drop table if exists fill_missing_values_test_input;
create table fill_missing_values_test_input(
  col_string string,
  col_bigint bigint,
  col_double double,
  col_boolean boolean,
  col_datetime datetime);
insert overwrite table fill_missing_values_test_input
select
*
from
(
  select
    '01' as col_string,
    10 as col_bigint,
    10.1 as col_double,
    True as col_boolean,
    cast('2016-07-01 10:00:00' as datetime) as col_datetime
  from dual
  union all
  select
    cast(null as string) as col_string,
    11 as col_bigint,
    10.2 as col_double,
    False as col_boolean,
    cast('2016-07-02 10:00:00' as datetime) as col_datetime
  from dual
  union all
  select
    '02' as col_string,
    cast(null as bigint) as col_bigint,
    10.3 as col_double,
    True as col_boolean,
```

```

    cast('2016-07-03 10:00:00' as datetime) as col_datetime
  from dual
 union all
 select
   '03' as col_string,
   12 as col_bigint,
   cast(null as double) as col_double,
   False as col_boolean,
   cast('2016-07-04 10:00:00' as datetime) as col_datetime
  from dual
 union all
 select
   '04' as col_string,
   13 as col_bigint,
   10.4 as col_double,
   cast(null as boolean) as col_boolean,
   cast('2016-07-05 10:00:00' as datetime) as col_datetime
  from dual
 union all
 select
   '05' as col_string,
   14 as col_bigint,
   10.5 as col_double,
   True as col_boolean,
   cast(null as datetime) as col_datetime
  from dual
) tmp;

```

- Input description

```

+-----+-----+-----+-----+-----+
| col_string | col_bigint | col_double | col_boolean | col_datetime |
+-----+-----+-----+-----+-----+
| 04 | 13 | 10.4 | NULL | 2016-07-05 10:00:00 |
| 02 | NULL | 10.3 | true | 2016-07-03 10:00:00 |
| 03 | 12 | NULL | false | 2016-07-04 10:00:00 |
| NULL | 11 | 10.2 | false | 2016-07-02 10:00:00 |
| 01 | 10 | 10.1 | true | 2016-07-01 10:00:00 |
| 05 | 14 | 10.5 | true | NULL |
+-----+-----+-----+-----+-----+

```

PAI command

```
drop table if exists fill_missing_values_test_input_output;
drop table if exists fill_missing_values_test_input_model_output;
PAI -name FillMissingValues
-project algo_public
-Dconfigs="col_double,null,mean;col_string,null-empty,str_type_empty;col_bigint,null,max;col_boolean,null,true;
col_datetime,null,2016-07-06 10:00:00"
-DoutputParaTableName="fill_missing_values_test_input_model_output"
-Dlifecycle="28"
-DoutputTableName="fill_missing_values_test_input_output"
-DinputTableName="fill_missing_values_test_input";
drop table if exists fill_missing_values_test_input_output_using_model;
drop table if exists fill_missing_values_test_input_output_using_model_model_output;
PAI -name FillMissingValues
-project algo_public
-DoutputParaTableName="fill_missing_values_test_input_output_using_model_model_output"
-DinputParaTableName="fill_missing_values_test_input_model_output"
-Dlifecycle="28"
-DoutputTableName="fill_missing_values_test_input_output_using_model"
-DinputTableName="fill_missing_values_test_input";
```

Output

- fill_missing_values_test_input_output

```
+-----+-----+-----+-----+-----+
| col_string | col_bigint | col_double | col_boolean | col_datetime |
+-----+-----+-----+-----+-----+
| 04 | 13 | 10.4 | true | 2016-07-05 10:00:00 |
| 02 | 14 | 10.3 | true | 2016-07-03 10:00:00 |
| 03 | 12 | 10.3 | false | 2016-07-04 10:00:00 |
| str_type_empty | 11 | 10.2 | false | 2016-07-02 10:00:00 |
| 01 | 10 | 10.1 | true | 2016-07-01 10:00:00 |
| 05 | 14 | 10.5 | true | 2016-07-06 10:00:00 |
+-----+-----+-----+-----+-----+
```

- fill_missing_values_test_input_model_output

```
+-----+-----+
| feature | json |
+-----+-----+
| col_string | {"name": "fillMissingValues", "type": "string", "paras":{"missing_value_type": "null-empty", "replaced_value": "str_type_empty"}} |
| col_bigint | {"name": "fillMissingValues", "type": "bigint", "paras":{"missing_value_type": "null", "replaced_value": 14}} |
| col_double | {"name": "fillMissingValues", "type": "double", "paras":{"missing_value_type": "null", "replaced_value": 10.3}} |
| col_boolean | {"name": "fillMissingValues", "type": "boolean", "paras":{"missing_value_type": "null", "replaced_value": 1}} |
| col_datetime | {"name": "fillMissingValues", "type": "datetime", "paras":{"missing_value_type": "null", "replaced_value": 1467770400000}} |
+-----+-----+
```

- fill_missing_values_test_input_output_using_model

```
+-----+-----+-----+-----+-----+
| col_string | col_bigint | col_double | col_boolean | col_datetime |
+-----+-----+-----+-----+-----+
| 04 | 13 | 10.4 | true | 2016-07-05 10:00:00 |
| 02 | 14 | 10.3 | true | 2016-07-03 10:00:00 |
| 03 | 12 | 10.3 | false | 2016-07-04 10:00:00 |
| str_type_empty | 11 | 10.2 | false | 2016-07-02 10:00:00 |
| 01 | 10 | 10.1 | true | 2016-07-01 10:00:00 |
| 05 | 14 | 10.5 | true | 2016-07-06 10:00:00 |
+-----+-----+-----+-----+-----+
```

- fill_missing_values_test_input_output_using_model_model_output

```
+-----+-----+
| feature | json |
+-----+-----+
| col_string | {"name": "fillMissingValues", "type": "string", "paras":{"missing_value_type": "null-empty", "replaced_value": "str_type_empty"}} |
| col_bigint | {"name": "fillMissingValues", "type": "bigint", "paras":{"missing_value_type": "null", "replaced_value": 14}} |
| col_double | {"name": "fillMissingValues", "type": "double", "paras":{"missing_value_type": "null", "replaced_value": 10.3}} |
| col_boolean | {"name": "fillMissingValues", "type": "boolean", "paras":{"missing_value_type": "null", "replaced_value": 1}} |
| col_datetime | {"name": "fillMissingValues", "type": "datetime", "paras":{"missing_value_type": "null", "replaced_value": 1467770400000}} |
+-----+-----+
```

4.4.3.3.4. Normalization

You can normalize one or more columns in a table and save the generated data to a new table.

Linear function transformation is supported. The transformation expression is $y = \frac{x - \text{MinValue}}{\text{MaxValue} - \text{MinValue}}$.

MaxValue and *MinValue* indicate the maximum and minimum values of the sample respectively.

- Click **Columns** to select the columns to be normalized. Double and bigint types are supported.
- You can choose whether to retain the original columns. If you select the corresponding checkbox, the original columns will be retained. Processed columns will be renamed.

PAI command

```
PAI -name normalize_wf
-project algo_public
-DkeepOriginal="true"
-DoutputTableName="test_4"
-DinputPartitions="pt=20150501"
-DinputTableName="bank_data_partition"
-DselectedColNames="emp_var_rate,euribor3m";
```

Algorithm parameters

Parameters

Parameter	Description	Default value
<code>inputTableName</code>	Required. The name of the input table.	N/A
<code>selectedColNames</code>	Optional. The names of columns selected from the input table.	All columns are selected by default.
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table for training.	The whole table is selected by default.
<code>outputTableName</code>	Required. The name of the output table.	N/A
<code>outputParaTableName</code>	Required. The name of the output configuration table.	N/A
<code>inputParaTableName</code>	Optional. The name of the input configuration table.	No input configuration table is set by default.
<code>outputPMMLTableName</code>	Required. The name of the output PMML table.	N/A
<code>keepOriginal</code>	Optional. This parameter specifies whether to retain the original columns. If <code>keepOriginal</code> is set to true, processed columns are renamed with the <code>normalized_</code> prefix and the original columns are retained and their data overwritten. If <code>keepOriginal</code> is set to false, all columns are retained but not renamed.	false
<code>lifecycle</code>	Optional. The lifecycle of the output table.	No lifecycle is set by default.
<code>coreNum</code>	Optional. The number of cores.	Automatically calculated.
<code>memSizePerCore</code>	Optional. The memory size of each core.	Automatically calculated.

4.4.3.3.5. Standardization

You can standardize one or more columns in a table and save the generated data to a new table.

- The formula used for standardization is $(X - \text{Mean}) / (\text{Standard deviation})$.
 - Mean: The mean of samples.
 - Standard deviation: The standard deviation of samples. This variable is used when samples are used to calculate the total deviation. To make the calculated value closer to the mean, you must moderately increase the calculated standard deviation by using the formula $\frac{1}{N-1}$.

- The formula for calculating the standard deviation of samples:

$$S = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (X_i - \bar{X})^2}$$

\bar{X} represents the mean of samples X_1, X_2, \dots, X_n .

- You can choose whether to retain the original columns. If you select the corresponding checkbox, the original columns will be retained. Processed columns will be renamed.
- Click **Columns** and select columns to be standardized. Double and bigint types are supported.

PAI command

```
PAI -name Standardize
-project algo_public
-DkeepOriginal="false"
-DoutputTableName="test_5"
-DinputTablePartitions="pt=20150501"
-DinputTableName="bank_data_partition"
-DselectedColNames="euribor3m,pdays"
```

Standardization component

Parameters for the two input ports are as follows:

- `inputTableName`: the name of the input table to be standardized.
- `inputParaTableName`: the name of the input configuration table that contains the parameters generated by the standardization node. You can use an input configuration table to apply the configuration parameters of one table to a new table.

Parameters for the two output ports are as follows:

- `outputTableName`: the name of the standardized output table.
- `outputParaTableName`: the name of the output parameter table, which can be applied to other datasets.

Standardization parameters

The corresponding algorithm parameter for Reserve Original Columns is `keepOriginal`.

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	-	-
<code>selectedColNames</code>	Optional. The names of columns selected from the input table.	-	All columns are selected by default.

Parameter	Description	Valid values	Default value
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions in the input table are selected by default.
outputTableName	Required. The name of the output table.	-	-
outputParaTableName	Required. The name of the output configuration table.	-	-
outputPartition	Optional. The partitions in the output table.	-	-
inputParaTableName	Optional. The name of the input configuration table.	-	No input configuration table is set by default.
keepOriginal	Optional. This parameter specifies whether to retain the original columns. If this parameter is set to true, the original columns are retained and the column name is suffixed with <code>_orig</code> .	true and false	false
lifecycle	Optional. The lifecycle of the output table.	-	No lifecycle is set by default.
coreNum	Optional. The number of cores.	-	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	-	Automatically calculated.

Examples

```
drop table if exists standardize_test_input;
create table standardize_test_input(
col_string string,
col_bigint bigint,
col_double double,
col_boolean boolean,
col_datetime datetime);
insert overwrite table standardize_test_input select * from (
select '01' as col_string,
10 as col_bigint,
10.1 as col_double,
True as col_boolean,
cast('2016-07-01 10:00:00' as datetime) as col_datetime from dual union all
select cast(null as string) as col_string,
11 as col_bigint,
10.2 as col_double,
False as col_boolean,
cast('2016-07-02 10:00:00' as datetime) as col_datetime from dual union all
select '02' as col_string,
cast(null as bigint) as col_bigint,
10.3 as col_double,
True as col_boolean,
cast('2016-07-03 10:00:00' as datetime) as col_datetime from dual union all
select '03' as col_string,
12 as col_bigint,
cast(null as double) as col_double,
False as col_boolean,
cast('2016-07-04 10:00:00' as datetime) as col_datetime from dual union all
select '04' as col_string,
13 as col_bigint,
10.4 as col_double,
cast(null as boolean) as col_boolean,
cast('2016-07-05 10:00:00' as datetime) as col_datetime from dual union all
select '05' as col_string,
14 as col_bigint,
10.5 as col_double,
True as col_boolean,
cast(null as datetime) as col_datetime from dual ) tmp;
```

PAI command

```

drop table if exists standardize_test_input_output;
drop table if exists standardize_test_input_model_output;
PAI -name Standardize
-project algo_public
-DoutputParaTableName="standardize_test_input_model_output"
-Dlifecycle="28"
-DoutputTableName="standardize_test_input_output"
-DinputTableName="standardize_test_input"
-DselectedColNames="col_double,col_bigint"
-DkeepOriginal="true";
drop table if exists standardize_test_input_output_using_model;
drop table if exists standardize_test_input_output_using_model_model_output;
PAI -name Standardize
-project algo_public
-DoutputParaTableName="standardize_test_input_output_using_model_model_output"
-DinputParaTableName="standardize_test_input_model_output"
-Dlifecycle="28"
-DoutputTableName="standardize_test_input_output_using_model"
-DinputTableName="standardize_test_input"
  
```

Input description

standardize_test_input

col_string	col_bigint	col_double	col_boolean	col_datetime
01	10	10.1	true	2016-07-01 10:00:00
NULL	11	10.2	false	2016-07-02 10:00:00
02	NULL	10.3	true	2016-07-03 10:00:00
03	12	NULL	false	2016-07-04 10:00:00
04	13	10.4	NULL	2016-07-05 10:00:00
05	14	10.5	true	NULL

Output description

standardize_test_input_output

col_string	col_bigint	col_double	col_boolean	col_datetime	stdized_col_bigint	stdized_col_double
01	10	10.1	true	2016-0	- 1.264911064	- 1.264911064
NULL	11	10.2	false	2016-07-02 10:00:00	- 0.632455532 0336759	- 0.632455532 0341972

col_string	col_bigint	col_double	col_boolean	col_datetime	stdized_col_bigint	stdized_col_double
02	NULL	10.3	true	2016-07-03 10:00:00	NULL	0.0
03	12	NULL	false	2016-07-04 10:00:00	0.0	NULL
04	13	10.4	NULL	2016-07-05 10:00:00	0.6324555320336759	0.6324555320341859
05	14	10.5	true	NULL	1.2649110640673518	1.2649110640683718

standardize_test_input_model_output

Feature	json
col_bigint	{"name": "standardize", "type": "bigint", "paras": {"mean": 12, "std": 1.58113883008419}}
col_double	{"name": "standardize", "type": "double", "paras": {"mean": 10.3, "std": 0.1581138830082909}}

standardize_test_input_output_using_model

col_string	col_bigint	col_double	col_boolean	col_datetime
01	-1.2649110640673515	-1.264911064068383	true	2016-07-01 10:00:00
NULL	-0.6324555320336758	-0.6324555320341971	false	2016-07-02 10:00:00
02	NULL	0.0	true	2016-07-03 10:00:00
03	0.0	NULL	false	2016-07-04 10:00:00
04	0.6324555320336758	0.6324555320341858	NULL	2016-07-05 10:00:00
05	1.2649110640673515	1.2649110640683716	true	NULL

standardize_test_input_output_using_model_model_output

feature	json
col_bigint	{"name": "standardize", "type": "bigint", "paras": {"mean": 12, "std": 1.58113883008419}}

feature	json
col_double	{"name": "standardize", "type": "double", "paras": {"mean": 10.3, "std": 0.1581138830082909}}

4.4.3.3.6. KV to Table

This component is used to convert KV pairs to a table. The key is converted to a table column, while the value is converted to a column value in the corresponding row.

KV table format definition:

- A key is the index of a column. Key values can be of the bigint or double types.
- A KV table can be input in the sparse format to algorithm components such as logistic and linear regression.
- Keys must be of the string type. You can input a key_map table to the KV to Table component to map keys to columns. This component outputs a key_map table that contains all key-column mappings after conversion, regardless of whether you input a key_map table.

kv
1:10;2:20;3:30

key_map table format definition: a table that contains index-to-column mappings and data type information. The data types of the col_name, col_index, and col_datatype columns must be string. The default data type of the col_datatype column is double if not specified.

col_name	col_index	col_datatype
col1	1	bigint
col2	2	double

PAI command

```
PAI -name KVtoTable
-project algo_public
-DinputTableName=test
-DoutputTableName=test_out
-DoutputKeyMapTableName=test_keymap_out
-DkvColName=kv;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	The table cannot be empty.
kvColName	Required. The name of the KV column.	Only one column can be selected.	-
outputTableName	Required. The name of the output table.	Table name	-

Parameter	Description	Valid values	Default value
outputKeyMapTableName	Required. The name of the output index table.	Table name	-
inputKeyMapTableName	Optional. The name of the input index table.	Table name	No input index table is set by default.
appendColName	Optional. The name of the appended column.	Multiple columns can be selected.	No column is appended by default.
inputTablePartitions	Optional. The partitions selected from the input table.	Partition name	No partition is specified by default.
kvDelimiter	Optional. The delimiter used to separate keys and values.	Symbol	The default delimiter is a semicolon (;).
itemDelimiter	Optional. The delimiter used to separate key-value pairs.	Symbol	The default delimiter is a comma (,).
top1200	Optional. This parameter specifies whether to output the first 1,200 columns.	true and false	Default value: true. If the value is false, an error is returned when the number of columns reaches the upper limit.
lifecycle	Optional. The lifecycle of the output table.	An integer greater than or equal to -1.	Default value: -1. This value indicates that no lifecycle is set.
coreNum	Optional. The number of cores.	An integer greater than 0.	Default value: -1. This value indicates that the number of instances is determined by the amount of input data.
memSizePerCore	Optional. The memory size of each core.	(100, 65536)	Default value: -1. This value indicates that the memory size is determined by the amount of input data.

Examples

SQL statement to generate data:

```
drop table if exists test;
create table test as
select
*
from
(
select '1:1,2:2,3:-3.3' as kv from dual
union all
select '1:10,2:20,3:-33.3' as kv from dual
) tmp;
```

PAI command

```
PAI -name KVToTable
-project algo_public
-DinputTableName=test
-DoutputTableName=test_out
-DoutputKeyMapTableName=test_keymap_out
-DkvColName=kv;
```

Output

The output table is shown as follows.

```
+-----+-----+-----+
| kv_1  | kv_2  | kv_3  |
+-----+-----+-----+
| 1.0   | 2.0   | -3.3   |
| 10.0  | 20.0  | -33.3  |
+-----+-----+-----+
```

The output mapping table is shown as follows.

```
+-----+-----+-----+
| col_name | col_index | col_type |
+-----+-----+-----+
| kv_1     | 1         | double   |
| kv_2     | 2         | double   |
| kv_3     | 3         | double   |
+-----+-----+-----+
```

Input and output restrictions

Converted columns include appended columns and columns converted from KV pairs. The KV columns are output before the appended columns. MaxCompute supports a maximum of 1,200 columns. When the number of columns exceeds the maximum value, and top1200 is set to true, only the first 1,200 columns are output. If top1200 is set to false, an error is returned. The number of input data entries cannot exceed 100 million.

Restrictions and guidelines

- If a key_map table is input, columns are converted from the keys that exist in both the key_map and key-value tables.
- The converted column type can only be numeric.
- If a key_map table is input, the data type of the converted key column is the same as that of the key_map table. If no key_map table is input, the data type of the converted key column is double.

- If a `key_map` table is not input, the name of the converted key column is in the format of 'kv column name+'+key'. An error is returned if the key contains any of the following characters: %&()*+-. /;<>=?
- If an appended column is specified and the name of the appended column is the same as that of the converted key column, an error is returned indicating a column name conflict.
- If a row contains multiple keys, the values are added.
- A column name can contain up to 128 characters. If more than 128 characters are entered, only the first 128 characters are kept.

4.4.3.3.7. Table to KV

This component is used to convert data tables to KV tables. Null values in the table to be converted are not displayed in the KV table. You can specify columns to be retained in the new table. These columns will remain unchanged.

PAI command

```
PAI -name TableToKV
  -project algo_public
  -DinputTableName=maple_tabletokv_basic_input
  -DoutputTableName=maple_tabletokv_basic_output
  -DselectedColNames=col0,col1,col2
  -DappendColNames=rowid;
```

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	Table name	-
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
<code>selectedColNames</code>	Optional. The names of selected columns from the input table.	The column type must be bigint or double.	The whole table is selected by default.
<code>appendColNames</code>	Optional. The names of columns to remain unchanged. These columns are written in the output table without any changes.	Multiple columns can be selected.	-

Parameter	Description	Valid values	Default value
outputTableName	Required. The name of the output KV table.	Table name	-
kvDelimiter	Optional. The delimiter used to separate keys and values.	Symbol	The default delimiter is a colon (:).
itemDelimiter	Optional. The delimiter used to separate key-value pairs.	Symbol	The default delimiter is a comma (,).
convertColToIndexId	Optional. This parameter specifies whether to convert columns into IDs.	0 and 1	0
inputKeyMapTableName	Optional. The name of the input index table. This parameter takes effect only in the case of <code>convertColToIndexId=1</code> . If this parameter is not specified, IDs are automatically generated.	Table name	No input index table is set by default.
outputKeyMapTableName	The name of the output index table. This parameter is required only in the case of <code>convertColToIndexId=1</code> .	Table name	The default value is determined by <code>convertColToIndexId</code> .
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	This parameter is used with <code>memSizePerCore</code> . The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

Example 1

Data generation

rowid	kv
0	col0:1,col1:1.1,col2:2
1	col0:0,col1:1.2,col2:3
2	col0:1,col1:2.3

rowid	kv
3	col0:1,col1:0.0,col2:4

PAI command

```
PAI -name TableToKV
  -project algo_public
  -DinputTableName=maple_tabletokv_basic_input
  -DoutputTableName=maple_tabletokv_basic_output
  -DselectedColNames=col0,col1,col2
  -DappendColNames=rowid;
```

Output

The output table is shown as follows.

maple_tabletokv_basic_output

rowid: bigint	kv: string
0	1:1.1,2:2
1	1:1.2,2:3
2	1:2.3
3	1:0.0,2:4

Example 2

PAI command

```
PAI -name TableToKV
  -project projectxlib4 -DinputTableName=maple_tabletokv_basic_input
  -DoutputTableName=maple_tabletokv_basic_output
  -DselectedColNames=col0,col1,col2 -DappendColNames=rowid
  -DconvertColToIndexId=1
  -DinputKeyMapTableName=maple_test_tabletokv_basic_map_input
  -DoutputKeyMapTableName=maple_test_tabletokv_basic_map_output;
```

Output

The output table is shown as follows.

maple_test_tabletokv_basic_map_output

col_name: string	col_index: string	col_datatype: string
col1	1	bigint
col2	2	double

Restrictions and guidelines

- If a key_map table is input, columns are converted from the keys that exist in both the key_map and key-value tables.

- If a key_map table is input and its type is different from the input table, the output key_map table uses the type specified by the user.
- The type of the columns that need to be converted into KV pairs in the input table must be bigint or double.

4.4.4. Feature engineering

4.4.4.1. Feature transformation

4.4.4.1.1. PCA

You can use principal component analysis (PCA) to reduce dimensionality.

- For more information about the PCA algorithm, see [Wikipedia](#).
- This component supports the dense data format.

PAI command

```
PAI -name PrinCompAnalysis
  -project algo_public
  -DinputTableName=bank_data
  -DeigOutputTableName=pai_temp_2032_17900_2
  -DprincompOutputTableName=pai_temp_2032_17900_1
  -DselectedColNames=pdays,previous,emp_var_rate,cons_price_idx,cons_conf_idx,euribor3m,nr_employed
  -DtransType=Simple
  -DcalcuType=CORR
  -DcontriRate=0.9;
```

Algorithm parameters

Parameters

Parameter	Description	Default value
inputTableName	Required. The name of the input table for PCA.	-
eigOutputTableName	Required. The name of the output table that contains eigenvectors and eigenvalues.	-
princompOutputTableName	Required. The name of the output table after PCA dimensionality reduction.	-
selectedColNames	Required. The names of feature columns that are involved in the PCA procedure.	-
transType	Optional. The method used to transform the original table to the principal component table. Valid values: Simple, Sub-Mean, and Normalization.	Simple
calcuType	Optional. The eigendecomposition mode of the original table. Valid values: CORR, COVAR_SAMP, and COVAR_POP.	CORR
contriRate	Optional. The ratio of information to be retained after dimensionality reduction.	0.9

Parameter	Description	Default value
remainColumns	Optional. The columns retained from the original table after dimensionality reduction.	-

Sample PCA output

Table after dimensionality reduction, shows a sample table after dimensionality reduction.

Eigenvalues and eigenvectors, shows the eigenvalues and eigenvectors.

4.4.4.2. Feature importance evaluation

4.4.4.2.1. Linear model feature importance

You can evaluate the quality of a linear algorithm model based on the predicted and actual output results such as the indicators and residual histogram. Indicators include SST, SSE, SSR, R2, R, MSE, RMSE, MAE, MAD, MAPE, count, yMean, and predictMean.

PAI command

```
pai -name regression_evaluation
  -project algo_public
  -DinputTableName=input_table
  -DyColName=y_col
  -DpredictionColName=prediction_col
  -DindexOutputTableName=index_output_table
  -DresidualOutputTableName=residual_output_table
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for training.	-	All partitions in the input table are selected by default.
yColName	Required. The name of the expected dependent variable column in the input table. It must be a numerical value.	-	-
predictionColName	Required. The name of the predicted dependent variable column. It must be a numerical value.	-	-
indexOutputTableName	Required. The name of the regression indicator output table.	-	-

Parameter	Description	Valid values	Default value
residualOutputTableName	Required. The name of the residual histogram output table.	-	-
intervalNum	Optional. The number of intervals to divide the histogram over.	-	100
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	-	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	-	Automatically calculated.

Output

The output table is in JSON format. [Field description](#) describes the JSON fields.

Field description

Field	Description
SST	Total sum of squares.
SSE	Sum of squared errors.
SSR	Sum of squares due to regression.
R2	Coefficient of determination.
R	Coefficient of multiple correlation.
MSE	Mean squared error.
RMSE	Root-mean-square error.
MAE	Mean absolute error.
MAD	Mean absolute difference.
MAPE	Mean absolute percentage error.
count	Number of rows.
yMean	Mean of expected dependent variables.
predictionMean	Mean of prediction results.

4.4.4.2.2. Random forest feature importance

You can calculate the importance of features in a random forest model.

Column settings

Fields Setting

Parameters Setting

Feature Columns Optional. ?

Select Column

Target Column Required.

PAI command

```

pai -name feature_importance
-p project algo_public
-DinputTableName=input
-DoutputTableName=output
-Dlabel=label
-DmodelName=model
    
```

Algorithm parameters

Parameters

Parameter	Description	Default value
inputTableName	Required. The name of the input table.	-
outputTableName	Required. The name of the output table.	-
labelColName	Required. The name of the label column.	-
modelName	Required. The name of the input model.	-
featureColNames	Optional. The names of feature columns selected from the input table.	All columns except the label column are selected by default.
inputTablePartitions	Optional. The partitions selected from the input table.	The whole table is selected by default.
lifecycle	Optional. The lifecycle of the output table.	No lifecycle is set by default.
coreNum	Optional. The number of cores.	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	Automatically calculated.

4.4.5. Statistical analysis

4.4.5.1. Data pivoting

This component allows you to view the distributions of feature values, feature columns, and label columns. Data can be analyzed more efficiently when you know its features. This component supports dense and sparse formats.

PAI command

```
PAI -name fe_meta_runner -project algo_public
-DinputTable="pai_dense_10_10"
-DoutputTable="pai_temp_2263_20384_1"
-DmapTable="pai_temp_2263_20384_2"
-DselectedCols="pdays,previous,emp_var_rate,cons_price_idx,cons_conf_idx,euribor3m,nr_employed,age,campaign,poutcome"
-DlabelCol="y"
-DcategoryCols="previous"
-Dlifecycle="28"-DmaxBins="5";
```

Algorithm parameters

Parameter	Description	Required	Default value
inputTable	The name of the input table.	Yes	N/A
inputTablePartitions	The partitions selected from the input table.	No	N/A
outputTable	The name of the output table.	Yes	N/A
mapTable	The output mapping table. The Data Pivoting component maps String and Int type data for machine learning to use for training.	Yes	N/A
selectedCols	The columns selected from the input table.	Yes	N/A
categoryCols	The columns specified to process Int or Double type columns as enumeration features.	No	null
maxBins	The maximum number of intervals for equal-distance division of continuous features.	No	100
isSparse	Indicates whether the features are in sparse format.	No	false

Parameter	Description	Required	Default value
itemSplitter	The delimiter used to separate sparse feature items.	No	","
kvSplitter	The delimiter used to separate keys and values.	No	"."
lifecycle	The lifecycle of the output table. Unit: days.	No	28

4.4.5.2. Whole table statistics

This component analyzes a table or selected columns of a table.

Parameter settings

In the Input Columns box, select the columns of the table to be analyzed. By default, all columns are selected. You can enter filtering conditions for the selected columns in the condition text box. Operators available include the equal (=), not equal (!=), greater than (>), less than (<), greater than or equal to (>=), and less than or equal to (<=) signs, as well as like and rlike.

PAI command

```
PAI -name SimpleSummary
  -project algo_public
  -DsummaryColNames="euribor3m,pdays"
  -DoutputTableNames="pai_temp_667_6017_1"
  -DinputTableName="bank_data"
  -Dfilter="age>40";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
summaryColNames	The columns that require analysis. Separate the columns with commas(,).
outputTableNames	The names of the output tables generated after the system performs the whole table statistics operation.
inputTableName	The name of the input table.
filter	The filtering conditions. Operators available include the equal (=), not equal (!=), greater than (>), less than (<), greater than or equal to (>=), and less than or equal to (<=) signs, as well as like and rlike.

4.4.5.3. Correlation coefficient matrix

The correlation coefficient is a measure of the correlation between columns in a matrix. The valid range of values for this parameter is [-1, 1]. The count equals the number of non-zero elements in two successive columns.

Column settings

Fields Setting
Tuning

All Selected by Default

Select Column

PAI command

```
PAI -name corrcoeff
-project algo_public
-DinputTableName=maple_test_corrcoeff_basic12x10_input
-DoutputTableName=maple_test_corrcoeff_basic12x10_output
-DcoreNum=1
-DmemSizePerCore=110
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartition	Optional. The partitions selected from the input table for training. in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions are selected by default.
outputTableName	Required. A list of output table names.	Table name	-
selectedColNames	Optional. The names of columns selected from the input table.	Column name	All columns are selected by default.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	This parameter is used with <code>memSizePerCore</code> . The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	Optional. The memory size of each node. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

Examples

Data generation describes the data generation result.

Data generation

col0:double	col1:bigint	col2:double	col3:bigint	col4:double	col5:bigint	col6:double	col7:bigint	col8:double	col9:double
19	95	33	52	115	43	32	98	76	40
114	26	101	69	56	59	116	23	109	105
103	89	7	9	65	118	73	50	55	81
79	20	63	71	5	24	77	31	21	75
87	16	66	47	25	14	42	99	108	57
11	104	38	37	106	51	3	91	80	97
84	30	70	46	8	6	94	22	45	48
35	17	107	64	10	78	53	34	90	96
13	61	39	1	29	117	112	2	82	28
62	4	102	88	100	36	67	54	12	85
49	27	44	93	68	110	60	72	86	58
92	119	0	113	41	15	74	83	18	111

PAI command

```
PAI -name corrcoeff
-project algo_public
-DinputTableName=maple_test_corrcoeff_basic12x10_input
-DoutputTableName=maple_test_corrcoeff_basic12x10_output
-DcoreNum=1
-DmemSizePerCore=110
```

Output description

Output table

columnsnames	col0	col1	col2
col0	1	-0.2115657251820724	0.0598306259706561
col1	-0.2115657251820724	1	-0.8444477377898585
col2	0.0598306259706561	-0.8444477377898585	1
col3	0.2599903570684693	-0.17507636221594533	0.18518346647293102
col4	-0.3483249188225586	0.40943384150571377	-0.20934839228057014

columnsnames	col0	col1	col2
col5	-0.28716254396809926	0.09135976026101403	-0.1896417512389659
col6	0.47880162127435116	-0.3018506374626574	0.1799377498863213
col7	-0.13646519484213326	0.40733726912808044	-0.3858885676469948
col8	-0.19500158764680092	-0.11827739124590071	0.20254569203773892
col9	0.3897390240949085	0.12433851389455183	0.13476160753756655

4.4.5.4. Covariance

In probability theory and statistics, covariance is a measure of the joint variability of two random variables. Variance is a special case of covariance where the two measured variables are the same. If the expected values are $E(X) = \mu$ and $E(Y) = \nu$, the covariance between real-number random variables X and Y is $\text{cov}(X, Y) = E((X - \mu)(Y - \nu))$.

PAI command

```
PAI -name cov
  -project algo_public
  -DinputTableName=maple_test_cov_basic12x10_input
  -DoutputTableName=maple_test_cov_basic12x10_output
  -DcoreNum=6
  -DmemSizePerCore=110;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
outputTableName	Required. A list of output table names.	Table name	-
selectedColNames	Optional. The names of columns selected from the input table.	Column name	All columns are selected by default.

Parameter	Description	Valid values	Default value
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	This parameter is used with memSizePerCore. The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	Optional. The memory size of each node. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

4.4.5.5. Empirical probability density chart

An empirical distribution is an estimated non-parametric distribution of probability in scenarios where accurate parametric distributions cannot be made.

The algorithm uses kernel distribution to estimate the probability density of sample data. Similar to a histogram, the algorithm generates functions to describe the distribution of sample data. However, kernel distribution is different in that it overlays the contributions of all parts to generate a smooth and continuous distribution curve, while a histogram only generates discrete descriptions. When kernel distribution is used, the probability density of non-sample data points is not 0, but an overlay of weighted probability density of all sampling points in a certain kernel distribution. In this document, the kernel distribution used is Gaussian distribution.

- For more information about kernel distribution, see [Wikipedia](#).
- For more information about empirical distribution, see [Wikipedia](#).

PAI command

```
PAI -name empirical_pdf
-project algo_public
-DinputTableName="test_data"
-DoutputTableName="test_epdf_out"
-DfeatureColNames="col0,col1,col2"
-DinputTablePartitions="ds='20160101'"
-Dlifecycle=1
-DintervalNum=100
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
outputTableName	Required. The name of the output table.	Table name	-
featureColNames	Required. The names of input columns.	Multiple columns of the double or bigint type can be selected.	-

Parameter	Description	Valid values	Default value
labelColName	Optional. The name of the input label column. The feature column is stratified based on the label values in the label column.	Only one column of the bigint or string type can be selected. The number of label values cannot exceed 100.	No input label column is set by default.
inputTablePartitions	Optional. The partitions selected from the input table.	Partition name	All partitions are selected by default.
intervalNum	The number of calculation intervals. The larger the number, the higher the accuracy.	[1, 1E14)	Default value: -1. This value indicates that the number of intervals is determined based on the range of data values for each column.
lifecycle	The lifecycle of the output table.	A positive integer	Default value: -1. This value indicates that no lifecycle is set.
coreNum	Optional. The number of cores.	A positive integer	Default value: -1. This value indicates that the number of instances is determined based on the volume of input data.
memSizePerCore	Optional. The memory size of each core.	A positive integer in the range of [1024, 65536]	Default value: -1. This value indicates that the memory size is determined based on the volume of input data.

Examples

SQL statement to generate data:

```
drop table if exists epdf_test;
create table epdf_test as
select
*
from
(
select 1.0 as col1 from dual
union all
select 2.0 as col1 from dual
union all
select 3.0 as col1 from dual
union all
select 4.0 as col1 from dual
union all
select 5.0 as col1 from dual
) tmp;
```

PAI command

```
PAI -name empirical_pdf
    -project algo_public
    -DinputTableName=epdf_test
    -DoutputTableName=epdf_test_out
    -DfeatureColNames=col1;
```

Input description

You can select multiple columns to be calculated. You can select a label column and stratify the columns by label. For example, if the label column contains labels 0 and 1, the columns that need to be calculated are stratified into two groups. One group only contains columns with label 0 and the other group only contains columns with label 1. The probability density for each group is then calculated. If no label column is selected, all feature columns are calculated.

Output description

This component outputs a diagram and a result table. The columns in the result table are as follows. If no label column is selected, NULL is output in the label column.

Column name	Data type
colName	string
label	string
x	double
pdf	double

Output table:

```
+-----+-----+-----+-----+
|colname |label  |x      |pdf    |
+-----+-----+-----+-----+
|col1    |NULL   |1.0    |0.12775155176809325 |
|col1    |NULL   |1.0404050505050506 |0.1304256933829622 |
|col1    |NULL   |1.0808101010101012 |0.13306325897429525 |
|col1    |NULL   |1.1212151515151518 |0.1356613897616418 |
|col1    |NULL   |1.1616202020202024 |0.1382173796574596 |
|col1    |NULL   |1.202025252525253  |0.1407286844875733 |
|col1    |NULL   |1.2424303030303037 |0.14319293014274642 |
|col1    |NULL   |1.2828353535353543 |0.14560791960033242 |
|col1    |NULL   |1.3232404040404049 |0.14797163876379316 |
|col1    |NULL   |1.3636454545454555 |0.1502822610772349 |
|col1    |NULL   |1.404050505050506  |0.1525381508819247 |
|col1    |NULL   |1.4444555555555567 |0.1547378654919243 |
|col1    |NULL   |1.4848606060606073 |0.1568801559764068 |
|col1    |NULL   |1.525265656565658  |0.15896396664681753 |
|col1    |NULL   |1.5656707070707085 |0.16098843325768245 |
|col1    |NULL   |1.6060757575757592 |0.1629528799404685 |
|col1    |NULL   |1.6464808080808098 |0.16485681490034038 |
|col1    |NULL   |1.6868858585858604 |0.16669992491584543 |
|col1    |NULL   |1.727290909090911  |0.16848206869138338 |
|col1    |NULL   |1.7676959595959616 |0.17020326912168932 |
|col1    |NULL   |1.8081010101010122 |0.17186370453638117 |
|col1    |NULL   |1.8485060606060628 |0.1734636990080946 |
|col1    |NULL   |1.8889111111111134 |0.17500371175692428 |
```

col1	NULL	1.9293161616161616	0.17648432589456017
col1	NULL	1.9697212121212121	0.17790623634938396
col1	NULL	2.0101262626262653	0.1792702373286898
col1	NULL	2.0505313131313131	0.18057720927022053
col1	NULL	2.0909363636363665	0.18182810544221673
col1	NULL	2.1313414141414141	0.18302393829491406
col1	NULL	2.1717464646464677	0.18416576567472337
col1	NULL	2.2121515151515183	0.1852546770123305
col1	NULL	2.252556565656569	0.18629177959496213
col1	NULL	2.2929616161616195	0.18727818503109434
col1	NULL	2.333366666666667	0.18821499601297229
col1	NULL	2.3737717171717208	0.18910329347850022
col1	NULL	2.4141767676767714	0.18994412426940221
col1	NULL	2.454581818181822	0.19073848937711185
col1	NULL	2.4949868686868726	0.19148733286168018
col1	NULL	2.535391919191923	0.1921915315221827
col1	NULL	2.575796969696974	0.19285188538972659
col1	NULL	2.6162020202020244	0.19346910910630113
col1	NULL	2.656607070707075	0.19404382424446043
col1	NULL	2.6970121212121256	0.1945765526142701
col1	NULL	2.7374171717171762	0.19506771059517916
col1	NULL	2.777822222222227	0.19551760452158667
col1	NULL	2.8182272727272775	0.19592642714194602
col1	NULL	2.858632323232328	0.1962942551623821
col1	NULL	2.8990373737373787	0.1966210478770638
col1	NULL	2.9394424242424293	0.1969066468790639
col1	NULL	2.979847474747478	0.19715077683721793
col1	NULL	3.0202525252525305	0.19735304731663747
col1	NULL	3.0606575757575781	0.19751295561309964
col1	NULL	3.1010626262626317	0.19762989056457925
col1	NULL	3.1414676767676823	0.19770313729675995
col1	NULL	3.1818727272727273	0.19773188285349683
col1	NULL	3.22227777777777836	0.19771522265793107
col1	NULL	3.262682828282834	0.19765216774530828
col1	NULL	3.303087878787885	0.19754165270453194
col1	NULL	3.3434929292929354	0.19738254426210697
col1	NULL	3.383897979797986	0.19717365043938664
col1	NULL	3.4243030303030366	0.19691373021193162
col1	NULL	3.4647080808080872	0.1966015035982942
col1	NULL	3.505113131313138	0.19623566210464843
col1	NULL	3.5455181818181885	0.19581487945135703
col1	NULL	3.585923232323239	0.19533782250778076
col1	NULL	3.6263282828282897	0.1948031623623475
col1	NULL	3.6667333333333403	0.1942095854560816
col1	NULL	3.707138383838391	0.19355580470939734
col1	NULL	3.7475434343434415	0.19284057057394655
col1	NULL	3.787948484848492	0.19206268194364004
col1	NULL	3.8283535353535427	0.19122099686158253
col1	NULL	3.8687585858585933	0.19031444296253852
col1	NULL	3.909163636363644	0.1893420275936375
col1	NULL	3.9495686868686946	0.18830284755928747
col1	NULL	3.989973737373745	0.1871960984396676
col1	NULL	4.030378787878796	0.18602108343567092
col1	NULL	4.070783838383846	0.18477722169674377
col1	NULL	4.111188888888897	0.1834640560916829
col1	NULL	4.151593939393948	0.1820812603860928
col1	NULL	4.191998989898998	0.18062864579383914

```

| col1 | NULL | 4.232404040404049 | 0.179106166873458 |
| col1 | NULL | 4.272809090909099 | 0.17751392674406796 |
| col1 | NULL | 4.31321414141415 | 0.17585218159888508 |
| col1 | NULL | 4.353619191919201 | 0.17412134449794325 |
| col1 | NULL | 4.394024242424251 | 0.1723219884250765 |
| col1 | NULL | 4.434429292929302 | 0.17045484859762067 |
| col1 | NULL | 4.4748343434343525 | 0.16852082402064342 |
| col1 | NULL | 4.515239393939403 | 0.1665209782808102 |
| col1 | NULL | 4.555644444444454 | 0.16445653957824907 |
| col1 | NULL | 4.596049494949504 | 0.16232889999798905 |
| col1 | NULL | 4.636454545454555 | 0.16013961402571825 |
| col1 | NULL | 4.6768595959596055 | 0.1578903963157465 |
| col1 | NULL | 4.717264646464656 | 0.15558311872216193 |
| col1 | NULL | 4.757669696969707 | 0.1532198066072439 |
| col1 | NULL | 4.798074747474757 | 0.1508026344442397 |
| col1 | NULL | 4.838479797979808 | 0.14833392073462115 |
| col1 | NULL | 4.878884848484859 | 0.14581612226291346 |
| col1 | NULL | 4.919289898989909 | 0.1432518277151203 |
| col1 | NULL | 4.95969494949496 | 0.1406437506896507 |
| col1 | NULL | 5.00010000000001 | 0.13799472213247665 |
+-----+-----+-----+-----+

```

Input and output restrictions

The maximum number of label columns that can be specified is 100.

4.4.5.6. Chi-square goodness of fit test

This component is used to determine the differences between the observed frequencies and the expected frequencies for each classification of a single multiclass classification nominal variable. The null hypothesis assumes that the observed frequencies and the expected frequencies are consistent.

PAI command

```

PAI -name chisq_test
  -project algo_public
  -DinputTableName=pai_chisq_test_input
  -DcolName=f0
  -DprobConfig=0:0.3,1:0.7
  -DoutputTableName=pai_chisq_test_output0
  -DoutputDetailTableName=pai_chisq_test_output0_detail

```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
colName	Required. The name of the column that requires a chi-square test.	Column name	-
outputTableName	Required. The name of the output table.	Table name that has not been used	-

Parameter	Description	Valid values	Default value
outputDetailTableName	Required. The name of the output detail table.	Table name that has not been used	-
inputTablePartitions	Optional. The partitions selected from the input table.	Partition list	All partitions are selected by default.
probConfig	Optional. The class probability configuration.	The configuration is stored in a key-value pair format: <code>class:probability</code> . The sum of all probabilities is 1.	All classes have the same probability by default.

Examples

Testing data

```
create table pai_chisq_test_input as
select * from
(
  select '1' as f0,'2' as f1 from dual
  union all
  select '1' as f0,'3' as f1 from dual
  union all
  select '1' as f0,'4' as f1 from dual
  union all
  select '0' as f0,'3' as f1 from dual
  union all
  select '0' as f0,'4' as f1 from dual
)tmp;
```

PAI command

```
PAI -name chisq_test
  -project algo_public
  -DinputTableName=pai_chisq_test_input
  -DcolName=f0
  -DprobConfig=0:0.3,1:0.7
  -DoutputTableName=pai_chisq_test_output0
  -DoutputDetailTableName=pai_chisq_test_output0_detail
```

Output description

Output table outputTableName is a JSON array containing only one row and one column.

```
{
  "Chi-Square": {
    "comment": "Pearsons chi-square test",
    "df": 1,
    "p-value": 0.75,
    "value": 0.2380952380952381
  }
}
```

Output table `outputDetailTableName` includes the following columns: data source class (`f0` or `f1`), observed frequency (`observed`), expected frequency (`expected`), and standard residuals (`residuals = (observed - expected)/sqrt(expected)`).

f0	f1	observed	expected	residuals
0	2	0.0	0.4	-0.6324555320336759
0	3	1.0	0.8	0.22360679774997894
0	4	1.0	0.8	0.22360679774997894
1	2	1.0	0.6000000000000001	0.5163977794943221
1	3	1.0	1.2000000000000002	-0.1825741858350555
1	4	1.0	1.2000000000000002	-0.1825741858350555

4.4.5.7. Chi-square test of independence

This component verifies whether two factors (each having two or more classes) are mutually independent. The null hypothesis is that two factors are independent of each other.

PAI command

```
PAI -name chisq_test
  -project algo_public
  -DinputTableName=pai_chisq_test_input
  -DxColName=f0
  -DyColName=f1
  -DoutputTableName=pai_chisq_test_output2
  -DoutputDetailTableName=pai_chisq_test_output2_detail
```

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	Table name	-
<code>xColName</code>	Required. The name of the column that requires a chi-square test.	Column name	-
<code>yColName</code>	Required. The name of the column that require a chi-square test.	Column name	-
<code>outputTableName</code>	Required. The name of the output table.	Table name that has not been used	-
<code>outputDetailTableName</code>	Required. The name of the output detail table.	Table name that has not been used	-
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table.	Partition list	All partitions are selected by default.

Parameter	Description	Valid values	Default value
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Examples

- Testing data

```
create table pai_chisq_test_input as
select * from
(
select '1' as f0,'2' as f1 from dual
union all
select '1' as f0,'3' as f1 from dual
union all
select '1' as f0,'4' as f1 from dual
union all
select '0' as f0,'3' as f1 from dual
union all
select '0' as f0,'4' as f1 from dual
)tmp;
```

- PAI command

```
PAI -name chisq_test
-project algo_public
-DinputTableName=pai_chisq_test_input
-DxColName=f0
-DyColName=f1
-DoutputTableName=pai_chisq_test_output2
-DoutputDetailTableName=pai_chisq_test_output2_detail
```

- Output description

Output table outputTableName is a JSON array containing only one row and one column.

```
{
  "Chi-Square": {
    "comment": "Pearsons chi-square test",
    "df": 2,
    "p-value": 0.75,
    "value": 0.8333333333333334
  }
}
```

Output table outputDetailTableName has the following columns:

Column name	Description
xColName	Class
yColName	Class
observed	Observed frequency
expected	Expected frequency

Column name	Description
residuals	Residuals = (observed - expected)/sqrt (expected)

Data:

f0	f1	observed	expected	residuals
0	2	0.0	0.4	-0.6324555320336759
0	3	1.0	0.8	0.22360679774997894
0	4	 1.0	0.8	0.22360679774997894
1	2	1.0	0.6000000000000001	0.5163977794943221
1	3	1.0	1.2000000000000002	-0.1825741858350555
1	4	1.0	1.2000000000000002	-0.1825741858350555

4.4.5.8. Scatter plot

In regression analysis, this component outputs a scatter plot that shows the distribution of data points in a Cartesian coordinate system.

Column settings

Fields Setting

Feature Columns Required. 

Select Column

Label Column Optional.



Samples Optional.

1000 

PAI command

```
PAI -name scatter_diagram
-project algo_public
-DselectedCols=emp_var_rate,cons_price_rate,cons_conf_idx,euribor3m
-DsampleSize=1000
-DlabelCol=y
-DmapTable=pai_temp_2447_22859_2
-DinputTable=scatter_diagram
-DoutputTable=pai_temp_2447_22859_1
```

Parameters

Parameter	Description	Default value
inputTable	Required. The name of the input table.	-

Parameter	Description	Default value
inputTablePartitions	Optional. The partitions selected from the input table.	-
outputTable	Required. The name of the output table that stores the samples.	-
mapTable	Required. The name of the output table that stores the maximum value, minimum value, and enumeration values of each feature.	-
selectedCols	Required. The columns selected from the input table from which to draw a scatter plot. A maximum of five features can be selected.	-
labelCol	Optional. An Int or String column to serve as the enumeration label column.	No enumeration label column is set by default.
sampleSize	Optional. The number of samples to collect from the input data.	1000
lifecycle	Optional. The lifecycle of the output table measured in days.	28

Examples

Input data

```
create table scatter_diagram as select emp_var_rate,cons_price_rate, cons_conf_idx,euribor3m,y from pai_bank_data limit 10
```

Parameters

emp_var_rate	cons_price_rate	cons_conf_idx	euribor3m	y
1.4	93.918	-42.7	4.962	0
-0.1	93.2	-42.0	4.021	0
-1.7	94.055	-39.8	0.729	1
-1.8	93.075	-47.1	1.405	0
-2.9	92.201	-31.4	0.869	1
1.4	93.918	-42.7	4.961	0
-1.8	92.893	-46.2	1.327	0
-1.8	92.893	-46.2	1.313	0
-2.9	92.963	-40.8	1.266	1
-1.8	93.075	-47.1	1.41	0
1.1	93.994	-36.4	4.864	0

emp_var_rate	cons_price_rate	cons_conf_idx	euribor3m	y
1.4	93.444	-36.1	4.964	0
1.4	93.444	-36.1	4.965	1
-1.8	92.893	-46.2	1.291	0
1.4	94.465	-41.8	4.96	0
1.4	93.918	-42.7	4.962	0
-1.8	93.075	-47.1	1.365	1
-0.1	93.798	-40.4	4.86	1
1.1	93.994	-36.4	4.86	0
1.4	93.918	-42.7	4.96	0
-1.8	93.075	-47.1	1.405	0
1.4	94.465	-41.8	4.967	0
1.4	93.918	-42.7	4.963	0
1.4	93.918	-42.7	4.968	0
1.4	93.918	-42.7	4.962	0
-1.8	92.893	-46.2	1.344	0
-3.4	92.431	-26.9	0.754	0
-1.8	93.075	-47.1	1.365	0
-1.8	92.893	-46.2	1.313	0
1.4	93.918	-42.7	4.961	0
1.4	94.465	-41.8	4.961	0
-1.8	92.893	-46.2	1.327	0
-1.8	92.893	-46.2	1.299	0
-2.9	92.963	-40.8	1.268	1
1.4	93.918	-42.7	4.963	0
-1.8	92.893	-46.2	1.334	0
1.4	93.918	-42.7	4.96	0
-1.8	93.075	-47.1	1.405	0
1.4	94.465	-41.8	4.96	0

emp_var_rate	cons_price_rate	cons_conf_idx	euribor3m	y
1.4	93.444	-36.1	4.962	0
1.1	93.994	-36.4	4.86	0
1.1	93.994	-36.4	4.857	0
1.4	93.918	-42.7	4.961	0
-3.4	92.649	-30.1	0.715	1
1.4	93.444	-36.1	4.966	0
-0.1	93.2	-42.0	4.076	0
1.4	93.444	-36.1	4.965	0
-1.8	92.893	-46.2	1.354	0
1.4	93.444	-36.1	4.967	0
1.4	94.465	-41.8	4.959	0
-1.8	92.893	-46.2	1.354	0
1.4	94.465	-41.8	4.958	0
-1.8	92.893	-46.2	1.354	0
1.4	94.465	-41.8	4.864	0
1.1	93.994	-36.4	4.859	0
1.1	93.994	-36.4	4.857	0
-1.8	92.893	-46.2	1.27	0
1.1	93.994	-36.4	4.857	0
1.1	93.994	-36.4	4.859	0
1.4	94.465	-41.8	4.959	0
1.1	93.994	-36.4	4.856	0
-1.8	93.075	-47.1	1.405	0
-1.8	92.843	-50.0	1.811	1
-0.1	93.2	-42.0	4.021	0
-2.9	92.469	-33.6	1.029	0
1.4	93.918	-42.7	4.962	0
-1.8	93.075	-47.1	1.365	0

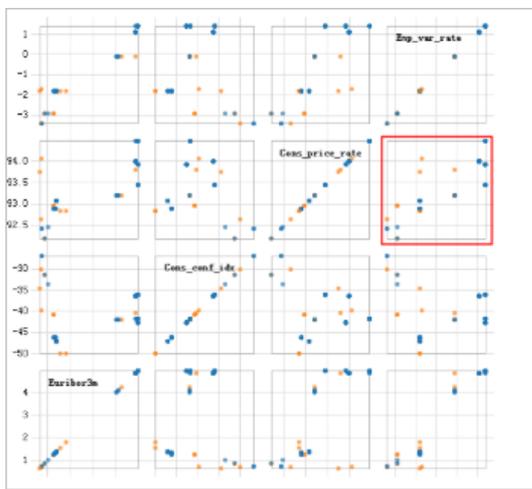
emp_var_rate	cons_price_rate	cons_conf_idx	euribor3m	y
1.1	93.994	-36.4	4.857	0
-1.8	92.893	-46.2	1.259	0
1.1	93.994	-36.4	4.857	0
1.4	94.465	-41.8	4.866	0
-2.9	92.201	-31.4	0.883	0
-0.1	93.2	-42.0	4.076	0
1.1	93.994	-36.4	4.857	0
1.4	93.918	-42.7	4.96	0
1.4	93.444	-36.1	4.962	0
1.1	93.994	-36.4	4.858	0
1.1	93.994	-36.4	4.857	0
1.1	93.994	-36.4	4.856	0
1.4	93.918	-42.7	4.968	0
1.4	93.444	-36.1	4.966	0
1.4	94.465	-41.8	4.962	0
1.4	93.444	-36.1	4.963	0
-1.8	92.843	-50.0	1.56	1
1.4	93.918	-42.7	4.96	0
1.4	93.444	-36.1	4.963	0
-3.4	92.431	-26.9	0.74	0
1.1	93.994	-36.4	4.856	0
1.4	93.918	-42.7	4.962	0
1.1	93.994	-36.4	4.856	0
-0.1	93.2	-42.0	4.245	1
1.1	93.994	-36.4	4.857	0
-1.8	93.075	-47.1	1.405	0
-1.8	92.893	-46.2	1.327	0
-0.1	93.2	-42.0	4.12	0

emp_var_rate	cons_price_rate	cons_conf_idx	euribor3m	y
1.4	94.465	-41.8	4.958	0
-1.8	93.749	-34.6	0.659	1
1.1	93.994	-36.4	4.858	0
1.1	93.994	-36.4	4.858	0
1.4	93.444	-36.1	4.963	0

Parameter settings

Scatter plot configuration: select `emp_var_rate`, `cons_price_rate`, `cons_conf_idx`, and `euribor3m` as the feature columns, and select `y` as the label column.

Output



You can view the distribution of classification tags between every two features in the scatter plot.

4.4.5.9. Two-sample T-test

A two-sample T-test is composed of an independent sample T-test and a paired sample T-test. Two samples independent of each other are called independent samples. An independent sample T-test checks whether two samples are significantly different from each other. The T-test is based on the premise that two samples are independent of each other and come from two normally distributed populations. A paired sample T-test checks whether the mean values from two paired populations are significantly different from each other.

PAI command

```
PAI -name t_test
-project algo_public
-DxTableName=pai_t_test_all_type
-DxColName=col1_double
-DxTablePartitions=ds=2010/dt=1
-DyTableName=pai_t_test_all_type
-DyColName=col1_double
-DyTablePartitions=ds=2010/dt=1
-DoutputTableName=pai_t_test_out
-Dalternative=less
-Dmu=47
-DconfidenceLevel=0.95
-Dpaired=False
-DvarEqual=True
```

Parameters

Parameter	Description	Valid values	Default value
xTableName	Required. The name of input table x.	Table name	-
xTablePartitions	Optional. The partitions selected from input table x for testing, in the format of Partition_name=value . To specify multiple partitions, use the following format: name1=value1/name2=value2 . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
xColName	Required. The column selected from table x for testing.	Column name. The type must be double or bigint.	-
yTableName	Required. The name of input table y.	Table name	-
yTablePartitions	Optional. The partitions selected from input table y for testing, in the format of Partition_name=value . To specify multiple partitions, use the following format: name1=value1/name2=value2 . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
yColName	Required. The name of the column selected from table y for testing.	Column name. The type must be double or bigint.	-
paired	Optional. A value of True indicates that it is a paired sample T-test. A value of False indicates that it is an independent sample T-test.	True and False	False
alternative	Optional. The alternative hypothesis.	two.sided, less, and greater	two.sided
mu	Optional. The hypothesized mean.	double	0
varEqual	Optional. This parameter indicates whether two population variances are equal.	True and False	False

Parameter	Description	Valid values	Default value
confidenceLevel	Optional. The confidence level.	0.8, 0.9, 0.95, 0.99, 0.995, and 0.999	0.95
coreNum	Optional. The number of cores.	This parameter is used with memSizePerCore. The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	Optional. The memory size of each node. Unit: MB.	A positive integer in the range of [1024, 65536].	Automatically calculated.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Custom resources

Note

- For a regular table, we recommend that you do not set coreNum and memSizePerCore, and instead allow the default values to be used automatically.
- If you do not have sufficient compute resources, use the following code to calculate the amount of compute resources needed:

```
def CalcCoreNumAndMem(row, centerCount, kOneCoreDataSize=1024):
    """Calculates the number of nodes and memory size needed for each node.
    Args:
        row: the number of rows in the input table.
        col: the number of columns in the input table.
        kOneCoreDataSize: the amount of data needed to be calculated per node. Unit: MB. The value must be a positive integer. Default value: 1024.
    Return:
        coreNum, memSizePerCore
    Example:
        coreNum, memSizePerCore = CalcCoreNumAndMem(1000,99, 100, kOneCoreDataSize=2048)
    """
    kMBytes = 1024.0 * 1024.0
    #Number of compute nodes
    coreNum = max(1, int((row * 2 * 8 / kMBytes) / kOneCoreDataSize))
    #Memory size per node = Data volume
    memSizePerCore = max(1024, int(kOneCoreDataSize*2))
    return coreNum, memSizePerCore
```

Examples

- SQL statement to generate data:

```
create table pai_test_input as
select * from
(
  select 1 as f0,2 as f1 from dual
  union all
  select 1 as f0,3 as f1 from dual
  union all
  select 1 as f0,4 as f1 from dual
  union all
  select 0 as f0,3 as f1 from dual
  union all
  select 0 as f0,4 as f1 from dual
)tmp;
```

- PAI command

```
PAI -name t_test
-project algo_public
-DxTableName=pai_test_input
-DxColName=f0
-DyTableName=pai_test_input
-DyColName=f1
-DyTablePartitions=ds=2010/dt=1
-DoutputTableName=pai_t_test_out
-Dalternative=less
-Dmu=47
-DconfidenceLevel=0.95
-Dpaired=False
-DvarEqual=True
```

- Output description

The output table is a JSON array containing only one row and one column.

```
{
  "AlternativeHypthesis": "difference in means not equals to 0",
  "ConfidenceInterval": "(-2.5465, -0.4535)",
  "ConfidenceLevel": 0.95,
  "alpha": 0.050000000000000004,
  "df": 19,
  "mean of the differences": -1.5,
  "p": 0.008000000000000007,
  "t": -3
}
```

Input and output restrictions

The input and output are not limited.

4.4.5.10. One-sample T-test

A one-sample T-test verifies whether the mean of a normally distributed population differs significantly from a target value. A T-test is performed based on the condition that the sample population is normally distributed.

PAI command

```
PAI -name t_test -project algo_public
-DxTableName=pai_t_test_all_type
-DxColName=col1_double
-DoutputTableName=pai_t_test_out
-DxTablePartitions=ds=2010/dt=1
-Dalternative=less
-Dmu=47
-DconfidenceLevel=0.95
```

Algorithm parameters

Parameter	Description	Valid values	Default value
xTableName	Required. The name of input table x.	Table name	-
xColName	Required. The column selected from table x for testing.	Column name. The type must be double or bigint.	-
outputTableName	Required. The name of the output table.	Table name that has not been used	-
xTablePartitions	Optional. The partitions selected from input table x.	Partition list	All partitions are selected by default.
alternative	Optional. The alternative hypothesis.	two.sided, less, and greater	two.sided
mu	Optional. The hypothesized mean.	double	0
confidenceLevel	Optional. The confidence level.	0.8, 0.9, 0.95, 0.99, 0.995, and 0.999	0.95

Output description

The output table is a JSON array containing only one row and one column.

```
{
  "AlternativeHypthesis": "mean not equals to 0",
  "ConfidenceInterval": "(44.72234194006504, 46.27765805993496)",
  "ConfidenceLevel": 0.95,
  "alpha": 0.05,
  "df": 99,
  "mean": 45.5,
  "p": 0,
  "stdDeviation": 3.919647479510927,
  "t": 116.081867662439
}
```

4.4.5.11. Lorenz curve

The Lorenz curve is a graph to illustrate the distribution of wealth across a population. The X axis represents the total population arranged from least wealthy to most wealthy, while the Y axis represents the total wealth. If this graph is a straight line, it indicates perfectly equal distribution of wealth. The Gini coefficient is calculated by taking the area between the equal distribution curve and the actual Lorenz curve for a population as a fraction of the total area beneath the equal distribution curve. As the distribution of wealth becomes less equal, the Gini coefficient will increase, whereas a population with equal distribution of wealth will have a Gini coefficient of 0.

To study the distribution of income among a population, American statistician Max Otto Lorenz proposed the famous Lorenz curve in 1905. In 1921, Italian economist Corrado Gini defined the Gini coefficient as a measure of inequality in a population based on the Lorenz curve.

PAI command

```
PAI -name LorenzCurve
    -project algo_public
    -DinputTableName=maple_test_lorenz_basic10_input
    -DcolName=col0
    -DoutputTableName=maple_test_lorenz_basic10_output -DcoreNum=20
    -DmemSizePerCore=110;
```

Parameters

Parameter	Description	Valid value	Default value
inputTableName	Required. The name of the input table.	Table name	N/A
outputTableName	Required. The name of the output table.	Table name that has not been used	N/A
colName	Optional. The column name. Separate multiple columns with commas (,).	Column name	The whole table is selected by default.
N	The number of quantiles.	N/A	100
inputPartitions	Optional. The partitions selected from the input table for training, in the <code>partition_name=value</code> format. To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	This parameter is used with <code>memSizePerCore</code> . The value must be a positive integer in the range of [1, 9999].	Automatically calculated.

Parameter	Description	Valid value	Default value
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

Examples

Data generation

col0:double
4
7
2
8
6
3
9
5
0
1
10

PAI command

```
PAI -name LorenzCurve
  -project algo_public
  -DinputTableName=maple_test_lorenz_basic10_input
  -DcolName=col0
  -DoutputTableName=maple_test_lorenz_basic10_output
  -DcoreNum=20
  -DmemSizePerCore=110;
```

Output

Quantile	col0
0	0
1	0.01818181818181818
2	0.01818181818181818
3	0.01818181818181818

Quantile	col0
4	0.01818181818181818
5	0.01818181818181818
6	0.01818181818181818
7	0.01818181818181818
8	0.01818181818181818
9	0.01818181818181818
10	0.01818181818181818
11	0.05454545454545454
12	0.05454545454545454
13	0.05454545454545454
14	0.05454545454545454
...	...
85	0.8181818181818182
86	0.8181818181818182
87	0.8181818181818182
88	0.8181818181818182
89	0.8181818181818182
90	1
91	1
92	1
93	1
94	1
95	1
96	1
97	1
98	1
99	1
100	1

4.4.5.12. Normality test

This component is used to determine whether observed values are normally distributed.

This component consists of three test methods: Anderson-Darling test (see [Wikipedia](#)), Kolmogorov-Smirnov test (see [Wikipedia](#)), and Q-Q plot (see [Wikipedia](#)). You can use one or more methods as needed.

Algorithm description:

- Original hypothesis H0: The observed values are normally distributed. H1: The observed values are not normally distributed.
- The KS p-value calculation method progressively calculates CDF of KS distribution regardless of the sample size. For more information, see [Wikipedia](#).
- If the sample size is greater than 1000, the Q-Q plot method collects samples to calculate and output plots. This means that the data points in plots do not necessarily cover all samples.

PAI command

```
PAI -name normality_test
  -project algo_public
  -DinputTableName=test
  -DoutputTableName=test_out
  -DselectedColNames=col1,col2
  -Dlifecycle=1;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
outputTableName	Required. The name of the output table.	Table name that has not been used	-
selectedColNames	Optional. The names of selected columns.	Multiple double or bigint type columns can be selected.	-
inputTablePartitions	Optional. The partitions selected from the input table.	Partition name	All partitions are selected by default.
enableQQplot	Optional. This parameter specifies whether to use the Q-Q plot.	true and false	true
enableADtest	Optional. This parameter specifies whether to perform the Anderson-Darling test.	true and false	true
enableKStest	Optional. This parameter specifies whether to perform the Kolmogorov-Smirnov test.	true and false	true

Parameter	Description	Valid values	Default value
lifecycle	Optional. The lifecycle of the output table.	An integer greater than or equal to -1	Default value: -1. This value indicates that no lifecycle is set.
coreNum	Optional. The number of cores.	An integer greater than 0	Default value: -1. This value indicates that the number of instances is determined by the amount of input data.
memSizePerCore	Optional. The memory size of each core.	(100, 65536)	Default value: -1. This value indicates that the memory size is determined by the amount of input data.

Examples

- SQL statement to generate data:

```
drop table if exists normality_test_input;
create table normality_test_input as
select
*
from
(
select 1 as x from dual
union all
select 2 as x from dual
union all
select 3 as x from dual
union all
select 4 as x from dual
union all
select 5 as x from dual
union all
select 6 as x from dual
union all
select 7 as x from dual
union all
select 8 as x from dual
union all
select 9 as x from dual
union all
select 10 as x from dual
) tmp;
```

- PAI command

```
PAI -name normality_test
-project projectxlib4
-DinputTableName=normality_test_input
-DoutputTableName=normality_test_output
-DselectedColNames=x
-Dlifecycle=1;
```

- **Input description**

Input format: select the columns that need to be calculated. The columns must be of the double or bigint type.

- **Output description**

A diagram and a result table are output. The columns in the result table are as follows. The result table has two partitions:

- `p='test'` shows the result of the AD or KS test. Data is output when `enableADtest` or `enableKStest` is set to true.
- `p='plot'` shows the O-Q plot data. When `enableQQplot` is set to true, data is output and the columns that meet the `p='test'` condition are reused. In the case of `p='plot'`, the `testvalue` column records the original observed data (x axis of the Q-Q plot), and the `pvalue` column records the expected data that is normally distributed (y axis of the Q-Q plot).

Output table:

```
+-----+-----+-----+-----+-----+
| colname | testname | testvalue | pvalue | p |
+-----+-----+-----+-----+-----+
|x | NULL | 1.0 | 0.8173291742279805 | plot |
|x | NULL | 2.0 | 2.470864450785345 | plot |
|x | NULL | 3.0 | 3.5156067948020056 | plot |
|x | NULL | 4.0 | 4.3632330349313095 | plot |
|x | NULL | 5.0 | 5.128868067945126 | plot |
|x | NULL | 6.0 | 5.871131932054874 | plot |
|x | NULL | 7.0 | 6.6367669650686905 | plot |
|x | NULL | 8.0 | 7.4843932051979944 | plot |
|x | NULL | 9.0 | 8.529135549214654 | plot |
|x | NULL | 10.0 | 10.182670825772018 | plot |
|x | Anderson_Darling_Test | 0.1411092332197832 | 0.9566579606430077 | test |
|x | Kolmogorov_Smirnov_Test | 0.09551932503797644 | 0.9999888659426232 | test |
+-----+-----+-----+-----+-----+
```

Column name	Data type	Definition
colName	string	Column name
testname	string	Test name
testvalue	double	Test value on the x axis of the Q-Q plot
pvalue	double	Test p value on the y axis of the Q-Q plot
p	double	Partition name

4.4.5.13. Percentile

This component calculates the percentile of the values in a column.

Parameter settings

Select the column to be analyzed. Only the double and bigint types are supported.

PAI command

```
PAI -name Percentile
-project algo_public
-DoutputTableName="pai_temp_666_6014_1"
-DcolName="euribor3m"
-DinputTableName="bank_data";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
outputTableName	The name of the output table automatically generated after the system performs the percentile calculation.
colName	The column selected for percentile calculation. Only the numeric type is supported.
inputTableName	The name of the input table.

4.4.5.14. Pearson coefficient

This component calculates the Pearson correlation coefficient of two numeric columns in an input table or a partition, and saves the result to the output table.

Component description

- The component has only two parameters: input column 1 and input column 2. Enter the names of the two columns for which the Pearson correlation coefficient is calculated.
- After you run the component, right-click the component and choose **View Analytics Report** from the short cut menu.
- The Pearson correlation coefficient is listed in the row.

PAI command

```
pai -name pearson
-project algo_test
-DinputTableName=wpbc
-Dcol1Name=f1
-Dcol2Name=f2
-DoutputTableName=wpbc_pear;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartitions	The partitions selected from the input table for calculation.	The parameter value must be in the <code>partition_name=value</code> format. To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	All partitions in the input table are selected by default.
col1Name	Required. The name of input column 1.	Column name	-
col2Name	Required. The name of input column 2.	Column name	-
outputTableName	Required. The name of the output table.	Table name	-

4.4.5.15. Histogram

This component analyzes data in a column and outputs a histogram.

Parameter settings

- Select the columns to be analyzed. Only the double and bigint types are supported.
- View the analysis report.
- You can adjust the step and move the slider to view the entire histogram.

4.4.6. Machine learning

4.4.6.1. Binary classification

4.4.6.1.1. GBDT binary classification

This component is used for binary classification based on GBDT regression and sorting. Values greater than the threshold value are considered positive samples, while values that are less than or equal to the threshold value are considered negative samples.

Procedure

1. Drag and drop the GBDT Binary Classification component onto the canvas for training and set the parameters, as shown in the following figure.

Fields Setting

Parameters Setting

Tuning

Feature Columns Double or bigint type. Max 8...

Select Column

Label Column Bigint type. Binary classification...

📁

Stratified Columns (optional) Double or bigint t...

📁

Parameters

Parameter	Description
Feature Columns	The double and bigint types are supported. A maximum of 800 columns can be specified.
Label Column	You can select all columns except the input column. The values must be of the binary type.
Stratification Column	Optional. The whole table is selected by default. The double and bigint types are supported.

- You can change the data type of the input columns.

The input columns of GBDT binary classification only support the continuous type and are processed in the same way as the discrete type.

- Set the parameters. Parameters

Parameter	Description
Metric Type	The normalized discounted cumulative gain (NDCG) and discounted cumulative gain (DCG).
Trees	Valid values: [1,10000]. Default value: 500.
Learning Rate	Valid values: (0, 1). Default value: 0.05.
Training Sample Fraction	Valid values: (0, 1). Default value: 0.6.
Training Feature Fraction	Valid values: (0, 1). Default value: 0.6.
Maximum Leaves	The value must be an integer in the range of [2, 1000]. Default value: 32.

Parameter	Description
Testing Data Fraction	Valid values: [0, 1]. Default value: 0.0.
Maximum Tree Depth	The value must be an integer in the range of [1, 11]. Default value: 11.
Minimum Samples per Leaf Node	The value must be an integer in the range of [100, 1000]. Default value: 500.
Random Seed	The value must be an integer in the range of [0, 10]. Default value: 0.
Maximum Splits per Feature	Valid values: [1, 1000]. Default value: 500.

4. View the output. For more information, see the description of the [Random forest](#) component.

 Note

- GBDT and GBDT_LR have different default types of loss functions. The default loss function of GBDT is regression loss: mean squared error loss. The default loss function of GBDT_LR is logistic regression loss. The system automatically writes the default loss function for GBDT_LR.
- For GBDT binary classification, the label column must be of the binary type. String type data is not supported.
- When connecting the ROC curve component, set the prediction component parameters and select a base value.

PAI command (F/L setup settings are not used)

```
PAI -name GBDT_LR
-project algo_public
-DfeatureSplitValueMaxSize="500"
-DrandSeed="0"
-Dshrinkage="0.5"
-DmaxLeafCount="32"
-DlabelColName="y"
-DinputTableName="bank_data_partition"
-DminLeafSampleCount="500"
-DgroupIDColName="nr_employed"
-DsampleRatio="0.6"
-DmaxDepth="11"
-DmodelName="xlab_m_GBDT_LR_21208"
-DmetricType="2"
-DfeatureRatio="0.6"
-DinputTablePartitions="pt=20150501"
-DtestRatio="0.0"
-DfeatureColNames="age,previous,cons_conf_idx,euribor3m"
-DtreeCount="500";
```

Parameters

Parameter	Description
name	The name of the component.

Parameter	Description
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
featureSplitValueMaxSize	Optional. The maximum number of splits per feature. Valid values: [1, 1000]. Default value: 500.
randSeed	Optional. The number of random seeds. The value must be an integer in the range of [0, 10]. Default value: 0.
shrinkage	Optional. The learning rate. Valid values: (0, 1). Default value: 0.05.
maxLeafCount	Optional. The maximum number of leaves. The value must be an integer in the range of [2, 1000]. Default value: 32.
labelColName	The name of the label column selected from the input table.
inputTableName	The name of the input table for training.
minLeafSampleCount	Optional. The minimum number of samples per leaf node. The value must be an integer in the range of [100, 1000]. Default value: 500.
groupIDColName	Optional. The name of the stratification column. The whole table is considered as a stratum by default.
sampleRatio	Optional. The fraction of samples collected for training. Valid values: (0, 1). Default value: 0.6.
maxDepth	Optional. The maximum depth of a tree. The value must be an integer in the range of [1, 11]. Default value: 11.
modelName	The name of the output model.
metricType	Optional. The type of a metric. Valid values: 0 and 1. 0 represents normalized discounted cumulative gain (NDCG) and 1 represents discounted cumulative gain (DCG).
featureRatio	Optional. The fraction of features collected for training. Valid values: (0, 1). Default value: 0.6.
inputTablePartitions	Optional. The partitions selected from the input prediction table. If no partitions are specified, the whole table is selected.
testRatio	Optional. The fraction of testing samples. Valid values: [0, 1]. Default value: 0.0.
featureColNames	The names of feature columns selected from the input table for training.
treeCount	Optional. The number of trees. Valid values: [1, 10000]. Default value: 500.

4.4.6.1.2. Linear SVM

Support-vector machines (SVMs) are developed based on the VC dimension theory and the structural risk minimization principle.

This linear SVM version is not implemented using the kernel function. For more information, see Trust Region Method for L2-SVM at <http://www.csie.ntu.edu.tw/~cjlin/papers/logistic.pdf>. This algorithm only supports binary classification.

Procedure

1. Configure column settings.
 - Feature Columns: You can select a feature column of the bigint or double type.
 - Label Column: The data type of the label column can be bigint, double, or string. This component only supports binary classification.
2. Set parameters. Parameters

Parameter	Description
Positive Sample Label	Optional. The value of the positive sample. If this parameter is not specified, the system randomly selects a value. We recommend that you specify this parameter when the positive and negative samples are significantly different.
Positive Penalty Factor	Optional. The weight of the positive sample. Valid values: (0, +∞). Default value: 1.0.
Negative Penalty Factor	Optional. The weight of the negative sample. Valid values: (0, +∞). Default value: 1.0.
Convergence Coefficient	Optional. The convergence deviation. Valid values: (0, 1). Default value: 0.001. <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note If no base value is specified, Positive Penalty Factor and Negative Penalty Factor must be set to the same value.</p> </div>

3. View the output. For more information, see the description of the [Random Forest](#) component.

PAI command (F/L setup settings are not used)

```
PAI -name LinearSVM
-project algo_public
-DnegativeCost="1.0"
-DmodelName="xlab_m_LinearSVM_6143"
-DpositiveCost="1.0"
-Depsilon="0.001"
-DlabelColName="y"
-DfeatureColNames="pdays,emp_var_rate,cons_conf_idx"
-DinputTableName="bank_data"
-DpositiveLabel="0";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.

Parameter	Description
negativeCost	Optional. The weight of the negative sample. It is the penalty factor of the negative sample. Valid values: (0, +∞). Default value: 1.0.
modelName	The name of the output model.
positiveCost	Optional. The weight of the positive sample. It is the penalty factor of the positive sample. Valid values: (0, +∞). Default value: 1.0.
epsilon	Optional. The convergence coefficient. Valid values: (0, 1). Default value: 0.001.
labelColName	The name of the label column.
featureColNames	The names of feature columns selected from the input table for training.
inputTableName	The name of the input table for training.
positiveLabel	Optional. The value of the positive sample. If this parameter is not specified, the system randomly selects a value.

4.4.6.1.3. Logistic regression for binary classification

Binary classification is a classic logistic regression method. Logistic regression on the algorithm platform supports multiclass classification. The logistic regression component supports two data types: sparse and dense.

Parameter settings

Parameters

Parameter	Description
Regularization Type	Optional. The type of regularization. Valid values: <i>L1</i> , <i>L2</i> , and <i>None</i> . Default value: <i>L1</i> .
Maximum Iterations	Optional. The maximum number of L-BFGS iterations. Default value: 100.
Regularization Coefficient	Optional. The regularization coefficient. Default value: 1.0. If regularizedType is set to <i>None</i> , this parameter is ignored.
Minimum Convergence Deviance	Optional. The condition to terminate L-BFGS. This is the log-likelihood deviation between two iterations. Default value: <i>1.0e-06</i> .

The logistic regression component outputs a model, which is available in the model list.

Model name format: `Experiment Name + "-" + Component Name + "model"` .

PAI command (F/L setup settings are not used)

```
PAI -name LogisticRegression
-project algo_public
-DmodelName="xlab_m_logistic_regression_6096"
-DregularizedLevel="1"
-DmaxIter="100"
-DregularizedType="l1"
-Depsilon="0.000001"
-DlabelColName="y"
-DfeatureColNames="pdays,emp_var_rate"
-DgoodValue="1"
-DinputTableName="bank_data";
```

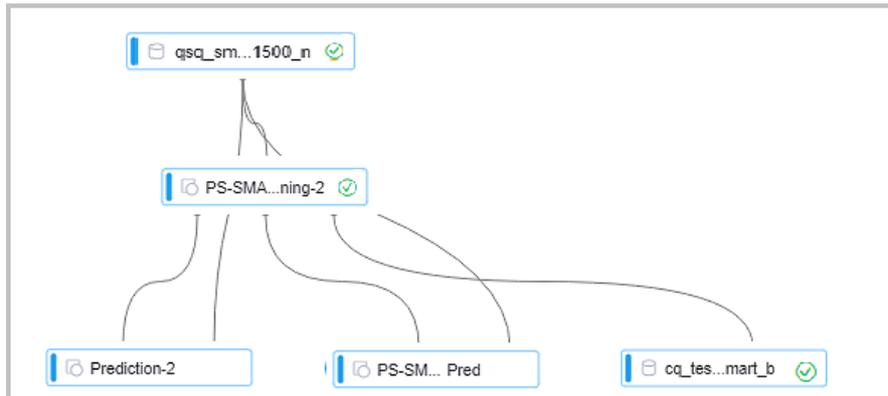
Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
modelName	The name of the output model.
regularizedLevel	Optional. The regularization coefficient. Default value: 1.0. If regularizedType is set to <i>None</i> , this parameter is ignored.
maxIter	Optional. The maximum number of L-BFGS iterations. Default value: 100.
regularizedType	Optional. The type of regularization. Valid values: <i>L1</i> , <i>L2</i> , and <i>None</i> . Default value: <i>L1</i> .
epsilon	Optional. The convergence deviation. It is the condition to terminate L-BFGS. This is the log-likelihood deviation between two iterations. Default value: <i>1.0e-06</i> .
labelColName	The name of the label column selected from the input table.
featureColNames	The names of feature columns selected from the input table for training.
goodValue	Optional. The base value. For binary classification, specify the label value of the training coefficient. If this parameter is not specified, the system randomly selects a value.
inputTableName	The name of the input table for training.

4.4.6.1.4. PS-SMART binary classification

A **parameter server** (PS) is used to train a large number of models online and offline. Scalable Multiple Additive Regression Tree (SMART) is an implementation of Gradient Boosting Decision Tree (GBDT) on PS. PS-SMART can run training tasks containing up to tens of billions of samples and hundreds of thousands of features on thousands of nodes. It also supports failover for high stability. PS-SMART supports various data formats, training targets, evaluation targets, output feature importance, and histogram approximation for training acceleration.

Quick start



As shown in the figure, a PS-SMART binary classification model is learned based on training data. The model has three output ports:

- Output model: off line model, which is connected to the unified prediction component. This model does not support the output of leaf node numbers.
- Output model table: a binary table that is not readable and is used to ensure compatibility with the PS-SMART prediction component. The table supports the output of leaf node numbers, which ensures higher efficiency, less resource consumption, and higher stability.
- Output feature importance table: lists the importance of each feature. Three importance types are supported. For more information, see [Parameters](#).

PAI command

- Training

```
PAI -name ps_smart
  -project algo_public
  -DinputTableName="smart_binary_input"
  -DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
  -DoutputTableName="pai_temp_24515_545859_2"
  -DoutputImportanceTableName="pai_temp_24515_545859_3"
  -DlabelColName="label"
  -DfeatureColNames="f0,f1,f2,f3,f4,f5"
  -DenableSparse="false"
  -Dobjective="binary:logistic"
  -Dmetric="error"
  -DfeatureImportanceType="gain"
  -DtreeCount="5";
  -DmaxDepth="5"
  -Dshrinkage="0.3"
  -Dl2="1.0"
  -Dl1="0"
  -Dlifecycle="3"
  -DsketchEps="0.03"
  -DsampleRatio="1.0"
  -DfeatureRatio="1.0"
  -DbaseScore="0.5"
  -DminSplitLoss="0"
```

- Prediction

```

PAI -name prediction
-project algo_public
-DinputTableName="smart_binary_input";
-DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
-DoutputTableName="pai_temp_24515_545860_1"
-DfeatureColNames="f0,f1,f2,f3,f4,f5"
-DappendColNames="label,qid,f0,f1,f2,f3,f4,f5"
-DenableSparse="false"
-Dlifecycle="28"
  
```

Parameters

- Data parameters

Command option	Parameter	Description	Valid values	Remarks
featureColNames	Feature Columns	The names of feature columns selected from the input table for training.	If the column name is in dense format, it must be of the bigint or double type. If the column name is in sparse KV format, it must be a string, and its keys and values must be numeric.	Required
labelColName	Label Column	The name of the label column selected from the input table.	The column name can be of either string or numeric type, but only numeric data can be stored in the columns. For example, in binary classification, the column value can be 0 or 1.	Required
weightCol	Weight Column	This column specifies the weight of each sample.	The column name can be of the numeric type.	Optional. Default value: null.
enableSparse	Use Sparse Format	This parameter specifies whether the data in the input table is in sparse format, in which key-value pairs are separated by spaces whereas keys and values are separated by colons (:), for example, 1:0.3 3:0.9.	[true, false]	Optional. Default value: false.

Command option	Parameter	Description	Valid values	Remarks
inputTableName	Input Table Name	N/A	N/A	Required
modelName	Output Model Name	N/A	N/A	Required
outputImportanceTableName	Output Feature Importance Table Name	N/A	N/A	Optional. Default value: null.
inputTablePartitions	Input Table Partitions	N/A	N/A	Optional. The parameter value must be in ds=1/pt=1 format.
outputTableName	Output Model Table Name	The output table is a MaxCompute table that uses the binary format and is not readable. The prediction component that comes with SMART can be used to generate leaf node numbers.	String	Optional
lifecycle	Output Table Lifecycle	N/A	Positive integer	Optional. Default value: 3.

- Algorithm parameters

Command option	Parameter	Description	Valid values	Remarks
objective	Objective Function Type	The objective function type affects learning and must be selected properly. Select binary:logistic for binary classification.	N/A	Required
metric	Evaluation Indicator Type	Evaluation indicators in the training set, which are exported to stdout of the coordinator in a logview.	logloss, error and auc	Optional. Default value: null.
treeCount	Trees	The number of trees. The training time is proportional to this number.	Positive integer	Optional. Default value: 1.

Command option	Parameter	Description	Valid values	Remarks
maxDepth	Maximum Tree Depth	The maximum depth of a tree. We recommend that you set this value to 5, which means the tree can contain up to 32 leaf nodes.	A positive integer in the range of [1, 20]	Optional. Default value: 5.
sampleRatio	Data Sampling Fraction	The data sampling rate when trees are built. The sample data is used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means data sampling is disabled.
featureRatio	Feature Sampling Fraction	The feature sampling rate when trees are built. The sample features are used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means feature sampling is disabled.
l1	L1 Penalty Coefficient	This parameter determines the number of leaf nodes. The greater the value, the less the leaf nodes. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 0.
l2	L2 Penalty Coefficient	This parameter determines the size of a leaf node. The greater the value, the more evenly the leaf nodes are distributed. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 1.0.
shrinkage	Learning Rate	N/A	(0, 1]	Optional. Default value: 0.3.

Command option	Parameter	Description	Valid values	Remarks
sketchEps	Sketch-based Approximate Precision	The threshold for selecting quantiles when you build a sketch. The number of buckets is $O(1.0/sketchEps)$. The smaller the parameter value, the more buckets are generated. Typically, you do not need to modify this value.	(0, 1)	Optional. Default value: 0.03.
minSplitLoss	Minimum Split Loss Change	The minimum split loss changes required for splitting a node. The greater the value, the more conservatively the node splits.	Non-negative real number	Optional. Default value: 0.
featureNum	Features	The number of features or the maximum feature ID. Specify this parameter for resource usage estimation.	Positive integer	Optional
baseScore	Global Offset	Original predicted values of all samples.	Real number	Optional. Default value: 0.5.
featureImportance Type	Feature Importance Type	The type of feature importance. weight indicates the number of times that a feature splits. gain indicates information gain brought by the feature. cover indicates the number of samples that the feature covers on the splitting nodes.	weight , gain , and cover	Optional. Default value: gain .

- Note

- Specify different values for the objective parameter in different learning models. On the binary classification Web GUI, the objective function is automatically specified and invisible to users. On the command line, set the objective parameter to `binary:logistic`.
- Mappings between metrics and objective functions are: **logloss** for **negative loglikelihood for logistic regression**, **error** for **binary classification error**, and **auc** for **Area under curve for classification**.

Execution optimization

Command option	Parameter	Description	Valid values	Remarks
coreNum	Cores	The number of cores. The greater the value, the faster the computing algorithm runs.	Positive integer	Optional. Automatically calculated.
memSizePerCore	Memory Size per Core (MB)	The memory size of each core, where 1024 represents 1 GB of memory.	Positive integer	Optional. Automatically calculated.

Example

- Data generation

The following example uses data in dense format.

```
drop table if exists lm_test_input;
create table smart_binary_input lifecycle 3 as
select
*
from
(
select 0.72 as f0, 0.42 as f1, 0.55 as f2, -0.09 as f3, 1.79 as f4, -1.2 as f5, 0 as label from dual
union all
select 1.23 as f0, -0.33 as f1, -1.55 as f2, 0.92 as f3, -0.04 as f4, -0.1 as f5, 1 as label from dual
union all
select -0.2 as f0, -0.55 as f1, -1.28 as f2, 0.48 as f3, -1.7 as f4, 1.13 as f5, 1 as label from dual
union all
select 1.24 as f0, -0.68 as f1, 1.82 as f2, 1.57 as f3, 1.18 as f4, 0.2 as f5, 0 as label from dual
union all
select -0.85 as f0, 0.19 as f1, -0.06 as f2, -0.55 as f3, 0.31 as f4, 0.08 as f5, 1 as label from dual
union all
select 0.58 as f0, -1.39 as f1, 0.05 as f2, 2.18 as f3, -0.02 as f4, 1.71 as f5, 0 as label from dual
union all
select -0.48 as f0, 0.79 as f1, 2.52 as f2, -1.19 as f3, 0.9 as f4, -1.04 as f5, 1 as label from dual
union all
select 1.02 as f0, -0.88 as f1, 0.82 as f2, 1.82 as f3, 1.55 as f4, 0.53 as f5, 0 as label from dual
union all
select 1.19 as f0, -1.18 as f1, -1.1 as f2, 2.26 as f3, 1.22 as f4, 0.92 as f5, 0 as label from dual
union all
select -2.78 as f0, 2.33 as f1, 1.18 as f2, -4.5 as f3, -1.31 as f4, -1.8 as f5, 1 as label from dual
) tmp;
```

- Training

Configure the training data and training components, as shown in [Quick start](#). Select the label column as the target column and columns f0, f1, f2, f3, f4, f5 as feature columns.

- You do not need to set the number of features because this number is calculated automatically by the algorithm. If you have a large number of features and want the algorithm to accurately estimate the amount of required resources, specify the actual number of features.
- To accelerate the training, set the number of cores on the execution optimization page. The greater the number, the faster the algorithm runs. Typically, you do not need to enter the memory size per core because the algorithm can accurately calculate the memory size. The PS algorithm starts to run only when all hosts have obtained the required resources. Therefore, you may need to wait for a longer period of time when the cluster is busy and resources are requested in large volumes.
- You can view the output values of the metrics in the stdout of the coordinator or in a logview (HTTP link starting with <http://logview.odps.aliyun-inc.com:8080/logview>). A single PS-SMART training job can contain multiple tasks, and therefore multiple logviews are created. Select the logview whose name starts with PS to view the output of the PS job.

- Prediction

- Use the unified prediction component

The model generated after training is saved in binary format and can be used for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#).

If the dense format is used, you only need to select feature columns. (All columns are selected by default, and extra columns do not affect the prediction.) If the KV format is used, set the data format to sparse format and select the correct delimiter. In the SMART model, key-value pairs are separated by space characters. Therefore, the delimiter must be set to space or `\u0020` (escape expression of spaces).

In the "prediction_detail" column, value 1 indicates a positive sample, and value 0 indicates a negative sample. The values following 0 and 1 indicate the probabilities of the corresponding classes.

- Use the PS-SMART prediction component

The output model table obtained after training is saved in binary format and can be used by the PS-SMART prediction component for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#). Set the required parameters, including the data format, feature columns, target column, and number of classes. The ID column can only be a string type column other than a feature column or a target column. The loss function must be set to `binary:logistic`.

The **prediction_score** column lists probabilities of predicted positive samples. A sample is predicted as a positive sample if its score is greater than 0.5. Otherwise, it is predicted as a negative sample. The **leaf_index** column lists the predicted leaf node numbers. Each sample has N numbers, where N is the number of decision trees. Each tree is mapped to a number, which indicates the leaf node number of the sample on this tree.

 Note

- The output model table is a binary table that is not readable and is used to support the PS-SMART prediction component. The table provides outputs such as leaf node numbers and evaluation indicators. However, the output table has strict requirements on data formats, which negatively affects user experience. This component is being continually improved, and may be replaced by another component in the future.
- A string type column must be selected as the label column. You can enter strings in the column but cannot be blank or NULL. A feature column can be converted to the string type by using the data type conversion component.
- The loss function must be explicitly set to `binary:logistic`. By default, the function does not work.

- View feature importance

To view feature importance, you can export the third output port to an output table, or right-click **PS-SMART training component** and choose **View Data > Output Feature Importance Table** from the shortcut menu.

order ▲	id ▲	value ▲
1	0	0.5690338015556335
2	1	0.21714292466640472
3	4	0.21382322907447815

In the table, the ID column lists the numbers of input features. In this example, the data is in dense format. The input features are **f0, f1, f2, f3, f4, f5**. Therefore, ID 0 represents f0 and ID 4 represents f4. If the KV format is used, the IDs represent keys in key-value pairs. Each value indicates a feature importance type. The default value is **gain**, indicating the sum of information gains brought by a feature in the model. The preceding figure shows only three features because only these three features are used during the tree split process. In this case, the importance of unused features is 0.

FAQ

- Q: Does PS_SMART support non-numerical features and tags?
- A: No.
- Q: What is the scale of features supported by PS-SMART? Can we use large-scale 0-1 features?
- A: Although PS-SMART supports tasks that contain hundreds of thousands of features, such tasks consume large amounts of resources and run slowly. Therefore, we recommend that you do not use such a large number of features. The GBDT algorithm is suitable for training with continuous features. The categorical features require one-hot coding to filter out infrequent features before they can be used for training. The continuous numerical features can be used for training with the GBDT algorithm directly. Discretization is not recommended for numerical features.
- Q: Why is the result different every time although the SMART algorithm has the same data and the same parameter settings?
- A: The PS-SMART algorithm applies randomness in many scenarios. For example, the `data_sample_ratio` and `fea_sample_ratio` items introduce data and feature sampling respectively. In addition, the PS-SMART algorithm uses histograms to show similarity. When multiple workers run in a cluster in distributed mode, local sketches are merged to global sketches in a random order. Although different merging orders result in different tree structures, this does not introduce too much variation to the output model. Therefore, it is normal situation to obtain different results after the algorithm runs multiple times with the same data and same parameter settings.

Note

- The target column in a PS-SMART binary classification model supports only numerical values (0 for negative samples and 1 for positive samples). Even if values in the MaxCompute table are strings, they are saved as numerical values. If the classification target is a type string similar to Good or Bad, convert it to 1 or 0.
- In the key-value format, feature IDs must be positive integers, and feature values must be real numbers. If feature IDs are strings, use the serialization component to serialize them. If the feature values are classification type strings, perform feature engineering, such as discretization.

4.4.6.2. Multiclass classification

4.4.6.2.1. KNN

The KNN algorithm is used to resolve classification issues. For a row in the prediction table, this component selects K entries nearest to the row from the training table. It then assigns the row to the class that is most common among the K entries.

PAI command

```
PAI -name knn
-DtrainTableName=pai_knn_test_input
-DtrainFeatureColNames=f0,f1
-DtrainLabelColName=class
-DpredictTableName=pai_knn_test_input
-DpredictFeatureColNames=f0,f1
-DoutputTableName=pai_knn_test_output
-Dk=2;
```

Parameters

Parameter	Description	Valid value	Default value
trainTableName	Required. The name of the training table.	Table name	N/A
trainFeatureColNames	Required. The names of feature columns selected from the training table.	Column name	N/A
trainLabelColName	Required. The name of the label column selected from the training table.	Column name	N/A
trainTablePartitions	Optional. The partitions selected from the training table.	Partition name	All partitions are selected by default.
predictTableName	Required. The name of the prediction table.	Table name	N/A
outputTableName	Required. The name of the output table.	Table name	N/A
predictFeatureColNames	Optional. The names of feature columns selected from the prediction table.	Column name	N/A
predictTablePartitions	Optional. The partitions selected from the prediction table.	Partition name	All partitions are selected by default.
appendColNames	Optional. The names of columns appended to the output table from the prediction table.	Column name	N/A
outputTablePartition	Optional. The partitions in the output table.	Partition name	The output table is non-partitioned by default.

Parameter	Description	Valid value	Default value
k	Optional. The number of the nearest neighbors.	A positive integer in the range of [1, 1000]	100
enableSparse	Optional. This parameter specifies whether the data in the input table is in sparse format.	true and false	false
itemDelimiter	Optional. The delimiter used to separate key-value pairs when the data in the input table is in sparse format.	Symbol	The default delimiter is a space.
kvDelimiter	Optional. The delimiter used to separate keys and values when the data in the input table is in sparse format.	Symbol	The default delimiter is a colon (:).
coreNum	Optional. The number of cores.	This parameter is used with memSizePerCore. The value must be a positive integer in the range of [1, 20000].	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Examples

- Test data

```
create table pai_knn_test_input as
select * from
(
  select 1 as f0,2 as f1, 'good' as class from dual
  union all
  select 1 as f0,3 as f1, 'good' as class from dual
  union all
  select 1 as f0,4 as f1, 'bad' as class from dual
  union all
  select 0 as f0,3 as f1, 'good' as class from dual
  union all
  select 0 as f0,4 as f1, 'bad' as class from dual
)tmp;
```

- PAI command

```
PAI -name knn
-DtrainTableName=pai_knn_test_input
-DtrainFeatureColNames=f0,f1
-DtrainLabelColName=class
-DpredictTableName=pai_knn_test_input
-DpredictFeatureColNames=f0,f1
-DoutputTableName=pai_knn_test_output
-Dk=2;
```

• Output description

f0	f1	prediction_result	prediction_score	prediction_detail
1	4	bad	1.0	{"bad": 1}
0	4	bad	1.0	{"bad": 1}
0	3	bad	0.5	{"bad": 0.5, "good": 0.5}
1	3	good	1.0	{"good": 1}
1	2	good	1.0	{"good": 1}

- o f0 and f1: the appended columns in the output table.
- o prediction_result: the classification result.
- o prediction_score: the probabilities for the classification result.
- o prediction_detail: the latest K conclusions and their probabilities.

4.4.6.2.2. Logistic regression for multiclass classification

Logistic regression of Apsara Stack Machine Learning Platform for AI supports multiclass classification. The logistic regression component supports two data formats: sparse and dense.

Parameter settings

Parameters

Parameter	Description
Regularization Type	Optional. The type of regularization. Valid values: <i>L1</i> , <i>L2</i> , and <i>None</i> . Default value: <i>L1</i> .
Max Iterations	Optional. The maximum number of L-BFGS iterations. Default value: 100.
Regularization Coefficient	Optional. The regularization coefficient. Default value: 1.0. If regularizedType is set to <i>None</i> , this parameter is ignored.
Minimum Convergence Deviance	Optional. The condition to terminate L-BFGS. This is the log-likelihood deviance between two iterations. Default value: <i>1.0e-06</i> .

The logistic regression component outputs a model, which is available in the model list.

Model naming format: Experiment Name + "-" + Component Name + "model" .

PAI command (F/L setup settings are not used)

```
PAI -name LogisticRegression
-project algo_public
-DmodelName="xlab_m_logistic_regression_6096"
-DregularizedLevel="1"
-DmaxIter="100"
-DregularizedType="l1"
-Depsilon="0.000001"
-DlabelColName="y"
-DfeatureColNames="pdays,emp_var_rate"
-DgoodValue="1"
-DinputTableName="bank_data";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
modelName	The name of the output model.
regularizedLevel	Optional. The regularization coefficient. Default value: 1.0. If regularizedType is set to <i>None</i> , this parameter is ignored.
maxIter	Optional. The maximum number of L-BFGS iterations. Default value: 100.
regularizedType	Optional. The type of regularization. Valid values: <i>L1</i> , <i>L2</i> , and <i>None</i> . Default value: <i>L1</i> .
epsilon	Optional. The convergence deviation. It is the condition to terminate L-BFGS. This is the loglikelihood deviation between two iterations. Default value: <i>1.0e-06</i> .
labelColName	The name of the label column selected from the input table.
featureColNames	The names of feature columns selected from the input table for training.
goodValue	Optional. The base value. For multiclass classification, specify the label value of the training coefficient. If this parameter is not specified, the system randomly selects a value.
inputTableName	The name of the input table for training.

4.4.6.2.3. Random forest

A random forest is a classifier that contains multiple decision trees. Its output class is decided by the mode of individual tree output classes.

Procedure

1. Drag and drop the Random Forest component onto the canvas and select columns. Parameters

Parameter	Description
-----------	-------------

Parameter	Description
Feature Columns	Optional. All columns except the label and weight columns are selected by default.
Excluded Columns	Optional. This parameter is used to exclude specified columns from training. This parameter is mutually exclusive with <code>featureColNames</code> .
Columns Forced to Convert	Optional. The default feature parsing rules are as follows: <ul style="list-style-type: none"> Parse columns of string, boolean, and datetime types to discrete columns. Parse columns of double and bigint types to contiguous columns. Set the <code>forceCategorical</code> parameter to parse bigint type columns to categorical columns.
Weight Columns	Optional. You can select all columns except the input and label columns. The double and bigint types are supported.
Label Column	You can select all columns except the input column. The bigint, double, and string types are supported.

2. Set the parameters of the Random Forest component. Parameters

Parameter	Description
Trees	The number of trees in the forest. Valid values: (0, 1000).
Single-tree Algorithm Type	Optional. The algorithm type of each tree in the forest. Valid values: <code>id3</code> , <code>c4.5</code> , and <code>cart</code> . If the forest has <code>n</code> trees and the condition is <code>algorithmTypes = a.b</code> , then <code>[0, a)</code> indicates <code>id3</code> , <code>[a, b)</code> indicates <code>cart</code> , and <code>[b, n)</code> indicates <code>c4.5</code> . If this parameter is set to <code>[2, 4]</code> for a forest with five trees, <code>[0, 1)</code> indicates the ID3 algorithm, <code>[2, 3)</code> indicates the CART algorithm, and <code>4</code> indicates the C4.5 algorithm. If the value is <code>None</code> , the algorithms are evenly allocated across the forest.
Random Features per Tree	The number of features selected randomly. Valid values: 1 to <code>N</code> . <code>N</code> indicates the number of features.
Minimum Samples per Leaf Node	Optional. The minimum number of samples per leaf node. The value must be a positive integer no less than 2.
Minimum Fraction of Samples on Leaf Node to Samples on Parent Node	The minimum fraction of samples on a leaf node to samples on a parent node. A value of <code>-1</code> indicates that no limit is set. Default value: <code>-1</code> . Valid values: <code>[0, 1]</code> .
Maximum Tree Depth	The maximum depth of a tree. <code>-1</code> indicates a completely grown tree. Valid values: <code>[1, ∞)</code> .
Random Samples Input per Tree	The number of random samples input per tree. Valid values: <code>(1000, 1000000]</code> .

 Note

- With improvement of the bagging method, the Random Forest component builds a forest without correlated trees in the big cube. Random forests are similar to the boosting method in many aspects, particularly their training processes.
- For the growth of a single tree, this method provides the id3, cart, and c4.5 options. The treeNum parameter is used to specify the number of trees in the forest, in the range of [1, 1000]. The structure of a single tree can be controlled based on the edited template. You can use other parameters to specify the minimum number of samples per leaf node, the minimum fraction of samples on a leaf node to samples on a parent node, and the maximum depth of a tree.
- Each row in the weight column corresponds to a sample and indicates the proportion of this sample in training. If the age column is selected as the weight column, the sample in the row with a higher weight value in the age column has a higher proportion during the training.
- The "input table is empty!" error may occur in the following situations: The sampling fraction is too small, which means that the value of maxRecordSize is too small, or the input table is empty.

PAI command (F/L setup settings are not used)

```
PAI -name RandomForest
-project algo_public
-DmodelName="xlab_m_random_forests_6036"
-DrandomColNum="1.0"
-DlabelColName="campaign"
-DmaxTreeDeep="10"
-DmaxRecordSize="100000"
-DfeatureColNames="age,pdays,previous,emp_var_rate,cons_price_idx,cons_conf_idx,euribor3m,nr_employed"
-DdisFeatureContinuous="1,1,1,1,1,1,1"
-DminNumPer="-1"
-DminNumObj="2"
-DinputTableName="bank_data"
-DweightColName="y"
-DtreeNum="10";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
modelName	The name of the output model.
randomColNum	Optional. The random attribute type. This parameter specifies the number of features randomly selected each time a single tree is generated. -7 indicates $\log_2 N$. Valid values: 1 to N. N indicates the number of features.
labelColName	The name of the label column selected from the input table.

Parameter	Description
maxTreeDeep	Optional. The maximum depth of a tree. <i>-1</i> indicates a completely grown tree. Valid values: $[1, \infty]$.
maxRecordSize	Optional. The maximum number of samples per tree. Valid values: (1000, 1000000). <i>-1</i> indicates 100000.
featureColNames	The names of feature columns selected from the input table for training.
isFeatureContinuous	Specifies whether the feature for subsequent columns is continuous or discrete. <i>1</i> indicates that the feature column data is continuous, while <i>0</i> indicates that the feature column data is discrete. <i>1,0,0</i> indicates that values are continuous in the first feature column and discrete in the second and third feature columns. The number of values corresponds to the feature length.
minNumPer	Optional. The minimum fraction of samples on a leaf node to samples on a parent node. A value of <i>-1</i> indicates that no limit is set. Valid values: $[0.0, 1.0]$.
minNumObj	Optional. The minimum number of samples per leaf node.
inputTableName	The name of the input table for training.
weightColName	Optional. The name of the weight column selected from the input table. If there is no weight column, set this parameter to <i>None</i> . If there is any weight column, the value of weightColName is greater than 0.
treeNum	The number of trees. Valid values: (0, 1000).

4.4.6.2.4. Naive Bayes

Naive Bayes classifier is a simple probabilistic classifier based on applying Bayes theorem with strong independence assumptions between the features. A probabilistic model that can more accurately describe this potential is called an independent feature model.

Component description

Component parameter settings (For more information, see the description of the Random Forest component.)

PAI command

```
PAI -name NaiveBayes
  -project algo_public
  -DmodelName="xlab_m_NaiveBayes_23772"
  -DinputTablePartitions="pt=20150501"
  -DlabelColName="poutcome"
  -DfeatureColNames="age,previous,cons_conf_idx,euribor3m"
  -DisFeatureContinuous="1,1,1,1"
  -DinputTableName="bank_data_partition";
```

Parameters

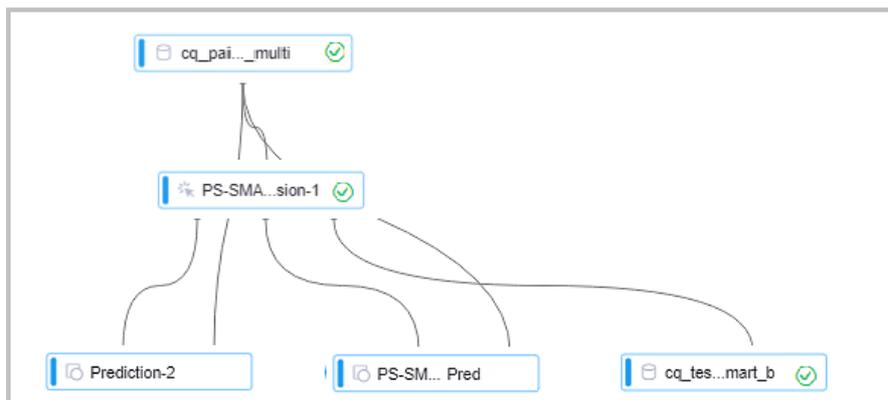
Parameter	Description
name	The name of the component.

Parameter	Description
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
modelName	The name of the model generated by training.
inputTablePartitions	Optional. The partitions selected from the input prediction table. If no partitions are specified, the entire table is selected.
labelColName	The name of the label column selected from the input table.
featureColNames	The names of feature columns selected from the input table for training.
isFeatureContinuous	Specifies whether the feature for subsequent columns is continuous or discrete. 1 indicates that the feature column data is continuous, while 0 indicates that the feature column data is discrete. 1,0,0 indicates that values are continuous in the first feature column and discrete in the second and third feature columns. The number of values corresponds to the feature length.
inputTableName	The name of the input table.

4.4.6.2.5. PS-SMART multiclass classification

A **parameter server** (PS) is used to train a large number of models online and offline. Scalable Multiple Additive Regression Tree (SMART) is an implementation of Gradient Boosting Decision Tree (GBDT) on PS. PS-SMART can run training tasks containing up to tens of billions of samples and hundreds of thousands of features on thousands of nodes. It also supports failover for high stability. PS-SMART supports various data formats, training targets, evaluation targets, output feature importance, and histogram approximation for training acceleration.

Quick start



As shown in the figure, a PS-SMART multiclass classification model is learned based on training data. The model has three output ports:

- Output model: offline model, which is connected to the unified prediction component. This model does not support the output of leaf node numbers.
- Output model table: a binary table that is not readable and is used to support the PS-SMART prediction component. The table provides outputs such as leaf node numbers and assessment metrics. However, the output table has strict requirements on data formats, which negatively affects user experience. This component is being continually improved, and may be replaced by another component in the future.

- Output feature importance table: lists the importance of each feature. Three importance types are supported. For more information, see [Parameters](#).

PAI command

- Training

```
PAI -name ps_smart
  -project algo_public
  -DinputTableName="smart_multiclass_input"
  -DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
  -DoutputTableName="pai_temp_24515_545859_2"
  -DoutputImportanceTableName="pai_temp_24515_545859_3"
  -DlabelColName="label"
  -DfeatureColNames="features"
  -DenableSparse="true"
  -Dobjective="multi:softprob"
  -Dmetric="mlogloss"
  -DfeatureImportanceType="gain"
  -DtreeCount="5";
  -DmaxDepth="5"
  -Dshrinkage="0.3"
  -DL2="1.0"
  -DL1="0"
  -Dlifecycle="3"
  -DsketchEps="0.03"
  -DsampleRatio="1.0"
  -DfeatureRatio="1.0"
  -DbaseScore="0.5"
  -DminSplitLoss="0"
```

- Prediction

```
PAI -name prediction
  -project algo_public
  -DinputTableName="smart_multiclass_input";
  -DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
  -DoutputTableName="pai_temp_24515_545860_1"
  -DfeatureColNames="features"
  -DappendColNames="label,features"
  -DenableSparse="true"
  -DkvDelimiter=":"
  -Dlifecycle="28"
```

Parameters

- Data parameters

Command option	Parameter	Description	Valid values	Remarks
----------------	-----------	-------------	--------------	---------

Command option	Parameter	Description	Valid values	Remarks
featureColNames	Feature Column	The names of feature columns selected from the input table for training.	If the column name is in dense format, it must be of the bigint or double type. If the column name is in sparse KV format, it must be a string, and its keys and values must be numeric.	Required
labelColName	Label Column	The name of the label column selected from the input table.	The column name can be of either string or numeric type, but only numeric data can be stored in the columns. For multiclass classification, column values can be 0, 1, 2, ..., n-1, where n is the number of classes.	Required
weightCol	Weight Column	This column specifies the weight of each sample.	The column name can be of the numeric type.	Optional. Default value: null.
enableSparse	Use Sparse Format	This parameter specifies whether the data in the input table is in sparse format, in which key-value pairs are separated by spaces whereas keys and values are separated by colons (:), for example, 1:0.3 3:0.9.	true, false	Optional. Default value: false.
inputTableName	Input Table Name	N/A	N/A	Required
modelName	Output Model Name	N/A	N/A	Required
outputImportanceTableName	Output Feature Importance Table Name	N/A	N/A	Optional. Default value: null.

Command option	Parameter	Description	Valid values	Remarks
inputTablePartitions	Input Table Partitions	N/A	N/A	Optional. The parameter value must be in ds=1/pt=1 format.
outputTableName	Output Model Table	The output table is a MaxCompute table that uses the binary format and is not readable. The prediction component that comes with SMART can be used to generate leaf node numbers.	String	Optional
lifecycle	Output Table Lifecycle	N/A	Positive integer	Optional. Default value: 3.

- Algorithm parameters

Command option	Parameter	Description	Valid values	Remarks
classNum	Classes	The number of classes in multiclass classification. If the number of classes is n, the label column name can be 0, 1, 2, ..., or n-1.	A non-negative integer, greater than or equal to 3.	Required
objective	Objective Function Type	The objective function type affects learning and must be selected properly. Set it to multi:softprob for multiclass classification.	N/A	Required
metric	Evaluation Indicator Type	Evaluation indicators in the training set, which are exported to stdout of the coordinator in a logview.	mloglossmerror	Optional. Default value: null.

Command option	Parameter	Description	Valid values	Remarks
treeCount	Trees	The number of trees. The training time is proportional to this number.	Positive integer	Optional. Default value: 1.
maxDepth	Maximum Decision Tree Depth	The maximum depth of a tree. We recommend that you set this value to 5, which means the tree can contain up to 32 leaf nodes.	A positive integer in the range of [1, 20].	Optional. Default value: 5.
sampleRatio	Data Sampling Fraction	The data sampling rate when trees are built. The sample data is used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means data sampling is disabled.
featureRatio	Feature Sampling Fraction	The feature sampling rate when trees are built. The sample features are used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means feature sampling is disabled.
l1	L1 Penalty Coefficient	This parameter determines the number of leaf nodes. The greater the value, the fewer the leaf nodes. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 0.
l2	L2 Penalty Coefficient	This parameter determines the size of a leaf node. The greater the value, the more evenly the leaf nodes are distributed. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 1.0.

Command option	Parameter	Description	Valid values	Remarks
shrinkage	Learning Rate	N/A	(0, 1]	Optional. Default value: 0.3.
sketchEps	Sketch-based Approximate Precision	The threshold for selecting quantiles when you build a sketch. The number of buckets is $O(1.0/sketchEps)$. The smaller the parameter value, the more buckets are generated. Typically, you do not need to modify this value.	(0, 1)	Optional. Default value: 0.03.
minSplitLoss	Minimum Split Loss	The minimum split loss changes required for splitting a node. The greater the value, the more conservatively the node splits.	Non-negative real number	Optional. Default value: 0.
featureNum	Features	The number of features or the maximum feature ID. Specify this parameter for resource usage estimation.	Positive integer	Optional
baseScore	Global Offset	Original predicted values of all samples.	Real number	Optional. Default value: 0.5.
featureImportance Type	Feature Importance Type	The type of feature importance. weight indicates the number of times that a feature splits. gain indicates information gain brought by the feature. cover indicates the number of samples that the feature covers on the splitting nodes.	weight , gain , and cover	Optional. Default value: gain .

- Note

- Specify different values for the objective parameter in different learning models. On the multiclass classification Web GUI, the objective function is automatically specified and invisible to users. On the command line, set the objective parameter to `multi:softprob`.
- Mappings between metrics and objective functions are: `mlogloss` for **multiclass negative log likelihood**, and `merror` for **multiclass classification error**.

- Execution optimization

Command option	Parameter	Description	Valid values	Remarks
coreNum	Cores	The number of cores. The greater the value, the faster the computing algorithm runs.	Positive integer	Optional. Automatically calculated.
memSizePerCore	Memory Size per Core (MB)	The memory size of each core, where 1024 represents 1 GB of memory.	Positive integer	Optional. Automatically calculated.

Example

- Data generation

The following example uses data in sparse KV format.

```
drop table if exists smart_multiclass_input;
create table smart_multiclass_input lifecycle 3 as
select
*
from
(
select 2 as label, '1:0.55 2:-0.15 3:0.82 4:-0.99 5:0.17' as features from dual
union all
select 1 as label, '1:-1.26 2:1.36 3:-0.13 4:-2.82 5:-0.41' as features from dual
union all
select 1 as label, '1:-0.77 2:0.91 3:-0.23 4:-4.46 5:0.91' as features from dual
union all
select 2 as label, '1:0.86 2:-0.22 3:-0.46 4:0.08 5:-0.60' as features from dual
union all
select 1 as label, '1:-0.76 2:0.89 3:1.02 4:-0.78 5:-0.86' as features from dual
union all
select 1 as label, '1:2.22 2:-0.46 3:0.49 4:0.31 5:-1.84' as features from dual
union all
select 0 as label, '1:-1.21 2:0.09 3:0.23 4:2.04 5:0.30' as features from dual
union all
select 1 as label, '1:2.17 2:-0.45 3:-1.22 4:-0.48 5:-1.41' as features from dual
union all
select 0 as label, '1:-0.40 2:0.63 3:0.56 4:0.74 5:-1.44' as features from dual
union all
select 1 as label, '1:0.17 2:0.49 3:-1.50 4:-2.20 5:-0.35' as features from dual
) tmp;
```

The data has five dimensions of features.

- Training

Configure the training data and training components, as shown in [Quick start](#). Select the label column as the target column and the features column as the feature column.

- You do not need to set the number of features because this number is calculated automatically by the algorithm. If you have a large number of features and want the algorithm to accurately estimate required resources, specify the actual number of features.
- To accelerate the training, set the number of cores on the execution optimization page. The greater the number, the faster the algorithm runs. Typically, you do not need to enter the memory size per core because the algorithm can accurately calculate the memory size. The PS algorithm starts to run only when all hosts have obtained the required resources. Therefore, you may need to wait for a longer time when the cluster is busy and resources are requested in large volumes.
- You can view the output values of the metrics in the stdout of the coordinator in a logview (HTTP link starting with <http://logview.odps.aliyun-inc.com:8080/logview>). A single PS-SMART training job can contain multiple tasks, which creates multiple logviews. Select the logview whose name starts with PS to view the output of the PS job.

Then, perform operations in the logview.

- Prediction

- Use the unified prediction component

The model generated after training is saved in binary format and can be used for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#).

If the dense format is used, you only need to select feature columns. (All columns are selected by default, and extra columns do not affect the prediction.) If the KV format is used, set the data format to sparse format and select the correct delimiter. In the SMART model, key-value pairs are separated with spaces. Therefore, the delimiter must be set to space or `\u0020` (escape expression of spaces).

In the `prediction_detail` column, values 0, 1, and 2 indicate classes, and the values following them indicate probabilities of the corresponding classes. The `predict_result` column lists the selected classes with the highest probability, and the `predict_score` column lists the probability of each selected class.

- Use the PS-SMART prediction component

The output model table obtained after training is saved in binary format and can be used by the PS-SMART prediction component for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#). Set the required parameters, including the data format, feature columns, target column, and number of classes. The ID column can only be a string type column other than a feature column or a target column. The loss function must be explicitly set to **multi:softprob**.

The **score_class_k** columns list probabilities of class k. The class with the highest probability is the predicted class. The **leaf_index** column lists the predicted leaf node numbers. Each sample has $N \times M$ numbers, where N is the number of decision trees, and M is the number of classes. In this example, each sample has 15 numbers ($5 \times 3 = 15$). Each tree is mapped to a number, which indicates the leaf node number of the sample on this tree.

Note

- The output model table is a binary table that is not readable and is used to support the PS-SMART prediction component. The table provides outputs such as leaf node numbers and evaluation indicators. However, the output table has strict requirements on data formats, which negatively affects user experience. This component is being continually improved, and may be replaced by another component in the future.
- A string type column must be selected as the label column. You can enter strings in the column but cannot be blank or NULL. A feature column can be converted to the string type by using the data type conversion component.
- The loss function must be explicitly set to **multi:softprob**. By default, the loss function does not work.

- View feature importance

To view feature importance, you can export the third output port to an output table, or right-click **PS-SMART training component** and choose **View Data > Output Feature Importance Table** from the shortcut menu. The following figure shows the output feature importance table.

order ▲	id ▲	value ▲
1	1	0.276059627532959
2	3	0.20854459702968597
3	4	0.31002077460289
4	5	0.20537501573562622

In the table, the ID column lists the numbers of input features. In this example, the data is in KV format, and the IDs represent keys in key-value pairs. If the dense format is used and input features are **f0,f1,f2,f3,f4,f5**, ID 0 represents f0 and ID 4 represents f4. Each value indicates a feature importance type. The default value is **gain**, indicating the sum of information gains brought by a feature in the model. The preceding figure shows only four features because only these four features are used during the tree split process. In this case, the importance of unused features is 0.

FAQ

- Q: Does PS_SMART support non-numerical features and tags?
- A: No.
- Q: What is the scale of features supported by PS-SMART? Can we use large-scale 0-1 features?
- A: Although PS-SMART supports tasks that contain hundreds of thousands of features, such tasks consume large amounts of resources and run slowly. Therefore, we recommend that you do not use a large number of features. The GBDT algorithm is suitable for training with continuous features. The categorical features require one-hot coding (to filter out infrequent features) before they can be used for training. The

continuous numerical features can be used for training with the GBDT algorithm directly. Discretization is not recommended for numerical features.

- Q: Why is the result different every time although the SMART algorithm has the same data and the same parameter settings?
- A: The PS-SMART algorithm applies randomness in many scenarios. For example, the `data_sample_ratio` and `fea_sample_ratio` items introduce data and feature sampling respectively. In addition, the PS-SMART algorithm uses histograms to show similarity. When multiple workers run in a cluster in distributed mode, local sketches are merged to global sketches in a random order. Although different merging orders result in different tree structures, this does not introduce too much variation to the output model. Therefore, it is normal situation to obtain different results after the algorithm runs multiple times with the same data and same parameter settings.

 Note

- The target column in a PS-SMART multiclass classification model supports only positive integer IDs (class numbers are 0, 1, 2, ..., n-1, where n is the number of classes). Even if the values in the MaxCompute table are strings, they are saved as numerical values. If the classification target is a type string similar to **Good**, **Medium**, or **Bad**, convert it into a numeric value (0, 1, 2, ..., n-1).
- In the key-value format, feature IDs must be positive integers, and feature values must be real numbers. If feature IDs are strings, use the serialization component to serialize them. If the feature values are classification type strings, perform feature engineering, such as discretization.

4.4.6.3. K-means clustering

K-means clustering is a widely used algorithm that is used to divide n objects into k clusters while maintaining high similarity within each cluster. Similarity is calculated based on the average value of objects in a cluster. This algorithm is similar to the expectation maximization algorithm for calculating mixed normality distribution, as both algorithms try to find the natural clustering center in data. K-means clustering randomly selects k objects. Each object represents the average value or center of a cluster. Based on its distance from each cluster center, each remaining object is then assigned to the nearest cluster and the average value of each cluster is re-calculated. This process is repeated until the criterion function converges. This algorithm assumes that object properties are from the spatial vector. Its objective is to minimize the sum of the mean square deviance inside each group.

Parameter settings

Parameters

Parameter	Description
Clusters	The number of clusters. Default value: 10.
Distance Measurement Method	Valid values: <i>euclidean</i> , <i>cityblock</i> (the sum of absolute deviations), and <i>cosine</i> . Default value: euclidean.
Initial Centroid Location	Valid values: <i>sample</i> (randomly selected), <i>topk</i> (first K rows), <i>uniform</i> (evenly distributed and randomly generated), <i>matrix</i> (an initial centroid table must be specified), and <i>kmpp</i> (k-means++ initialization). Default value: sample.
Maximum Iterations	The maximum number of iterations. Default value: 100.
Minimum Iteration Precision	The minimum iteration precision. Default value: 0.0.

Procedure

1. After running the K-means Clustering component, you can view the cluster center table.
 Cluster center table: The number of columns in this table is equal to the total number of columns selected from the input table. The number of rows is equal to the number of clusters, with each row representing a cluster center location.
2. Right-click the target table and choose **View Data** to view the cluster index table (idxTablename).
 - o Cluster index table: The number of rows is equal to the total number of rows in the input table. The value in each row represents the cluster index of the point in the corresponding row of the input table.
 - o The names of all columns are displayed. A classification marking column is appended to the table.
 - o 0, 1, 2, 3 are classification IDs.
 - o You can also use the table name generated by PAI command to view the cluster center table, cluster index table, and cluster count table in IDE.

 **Note** If *matrix* is selected as the initial centroid location, you must define the initial centroid table, with the same columns as the original table. The number of rows is the same as the number of clusters. When you prepare the table, configure k centers and use SQL or MapReduce for sampling, or select another method based on your requirements.

PAI command

```
PAI -name KMeans
-project algo_public
-DcenterCount="10"
-DidxTableName="bank_data_index"
-DdistanceType="euclidean"
-DappendColsIndex="0,1,2,3,4,5,6,7,8,9,10"
-DcenterTableName="pai_temp_3300_27298_3"
-Dloop="100"
-DclusterCountTableName="pai_temp_3300_27298_2"
-DinitCentersMethod="sample"
-Daccuracy="0.0"
-DinputTableName="bank_data"
-DselectedColNames="cons_conf_idx,emp_var_rate,euribor3m,pdays,previous";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
centerCount	The number of clusters. The value must be an integer. Default value: 10.
idxTableName	The name of the output cluster index table. The number of rows is equal to the total number of rows in the input table. The value in each row represents the cluster index of the point in the corresponding row of the input table.
distanceType	Optional. The method used to measure the distance. Valid values: euclidean, cityblock, and cosine. Default value: euclidean.

Parameter	Description
<code>appendColsIndex</code>	Optional. The name of the ID column appended to the output table. No ID column is appended to the output table by default.
<code>centerTableName</code>	The name of the output cluster center table. The number of columns in this table is equal to the total number of columns selected from the input table. The number of rows is equal to the number of clusters, with each row representing a cluster center location.
<code>loop</code>	Optional. The maximum number of iterations. The value must be an integer. Default value: 100.
<code>clusterCountTableName</code>	The name of the cluster point count table. The number of rows is equal to the number of clusters, which indicates the total number of cluster points in the class in each clustering centroid row.
<code>initCentersMethod</code>	Optional. The method used to determine the initial centroid location. The options include <code>sample</code> (randomly selected), <code>topk</code> (first K rows), <code>uniform</code> (evenly distributed and randomly generated), <code>matrix</code> (an initial centroid table must be specified), and <code>kmpp</code> (k-means++ initialization). Default value: <code>sample</code> .
<code>accuracy</code>	The minimum iteration precision. Default value: 0.0.
<code>inputTableName</code>	The name of the input table.
<code>selectedColNames</code>	The names of columns selected from the input table, which are separated with commas (.). Only double type is supported.
<code>initCenterTableName</code>	Optional. The name of the table that stores the initial center values. This table is not required unless <code>initCentersMethod</code> is set to <code>matrix</code> .

4.4.6.4. Regression

4.4.6.4.1. GBDT regression

Gradient boosting decision tree (GBDT) is an iterative decision tree algorithm based on multiple decision trees. The final output is the sum of conclusions of all trees. GBDT can be applied to almost all regression models (linear or nonlinear) and has a wider scope of application than logistic regression that is only applicable to linear regression.

For more information, see [A Regression Framework for Learning Ranking Functions Using Relative Relevance Judgments](#). For more information, see [GBDT binary classification](#).

PAI command

```

PAI -name gbdt
-project algo_public
-DfeatureSplitValueMaxSize="500"
-DlossType="0"
-DrandSeed="0"
-DnewtonStep="0"
-Dshrinkage="0.05"
-DmaxLeafCount="32"
-DlabelColName="campaign"
-DinputTableName="bank_data_partition"
-DminLeafSampleCount="500"
-DsampleRatio="0.6"
-DgroupIDColName="age"
-DmaxDepth="11"
-DmodelName="xlab_m_GBDT_83602"
-DmetricType="2"
-DfeatureRatio="0.6"
-DinputTablePartitions="pt=20150501"
-Dtau="0.6"
-Dp="1"
-DtestRatio="0.0"
-DfeatureColNames="previous,cons_conf_idx,euribor3m"
-DtreeCount="500"
  
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
featureColNames	Optional. The names of feature columns selected from the input table for training.	Column name	All columns are selected by default.
labelColName	Optional. The name of the label column selected from the input table.	Column name	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions are selected by default.
modelName	Required. The name of the output model.	-	-

Parameter	Description	Valid values	Default value
outputImportanceTableName	Optional. The name of the output feature importance table.	-	-
groupIDColName	Optional. The name of the stratification column.	Column name	The whole table is selected by default.
lossType	Optional. The loss function type. The function types include 0: <i>GBRANK</i> , 1: <i>LAMBDA MART_DCG</i> , 2: <i>LAMBDA MART_NDCG</i> , 3: <i>LEAST_SQUARE</i> , and 4: <i>LOG_LIKELIHOOD</i> .	0, 1, 2, 3, and 4	0
metricType	Optional. The type of metrics. 0 (<i>NDCG</i>) indicates the normalized discounted cumulative gain, 1 (<i>DCG</i>) indicates the discounted cumulative gain, and 2 (<i>AUC</i>) is applicable only to 0/1 label.	0, 1, and 2	2
treeCount	Optional. The number of trees.	[1, 10000]	500
shrinkage	Optional. The learning rate.	(0, 1]	0.05
maxLeafCount	Optional. The maximum number of leaves. This value must be an integer.	[2, 1000]	32
maxDepth	Optional. The maximum depth of a tree. This value must be an integer.	[1, 11]	11
minLeafSampleCount	Optional. The minimum number of samples on a leaf node. This value must be an integer.	[100, 1000]	500
sampleRatio	Optional. The fraction of training samples.	(0, 1]	0.6
featureRatio	Optional. The fraction of training features.	(0, 1]	0.6
tau	Optional. The Tau parameter in gbrank loss.	[0, 1]	0.6
p	Optional. The p parameter in gbrank loss.	[1, 10]	1

Parameter	Description	Valid values	Default value
randSeed	Optional. The random seed.	[0, 10]	0
newtonStep	Optional. This parameter specifies whether to use the Newton method.	0 and 1	1
featureSplitValueMaxSize	Optional. The maximum number of splits per feature.	[1, 1000]	500
lifecycle	Optional. The lifecycle of the output table.	-	No lifecycle is set by default.

Examples

SQL statement to generate data:

```
drop table if exists gbdt_ls_test_input;
create table gbdt_ls_test_input as select * from (
select  cast(1 as double) as f0,
        cast(0 as double) as f1,
        cast(0 as double) as f2,
        cast(0 as double) as f3,
        cast(0 as bigint) as label from dual union all
select  cast(0 as double) as f0,
        cast(1 as double) as f1,
        cast(0 as double) as f2,
        cast(0 as double) as f3,
        cast(0 as bigint) as label from dual union all
select  cast(0 as double) as f0,
        cast(0 as double) as f1,
        cast(1 as double) as f2,
        cast(0 as double) as f3,
        cast(1 as bigint) as label from dual union all
select  cast(0 as double) as f0,
        cast(0 as double) as f1,
        cast(0 as double) as f2,
        cast(1 as double) as f3,
        cast(1 as bigint) as label from dual union all
select  cast(1 as double) as f0,
        cast(0 as double) as f1,
        cast(0 as double) as f2,
        cast(0 as double) as f3,
        cast(0 as bigint) as label from dual union all
select  cast(0 as double) as f0,
        cast(1 as double) as f1,
        cast(0 as double) as f2,
        cast(0 as double) as f3,
        cast(0 as bigint) as label from dual ) a;
```

PAI command

- Training:

```
drop offlinemodel if exists gbdt_ls_test_model;
PAI -name gbdt
-project algo_public
-DfeatureSplitValueMaxSize="500"
-DlossType="3"
-DrandSeed="0"
-DnewtonStep="1"
-Dshrinkage="0.5"
-DmaxLeafCount="32"
-DlabelColName="label"
-DinputTableName="gbdt_ls_test_input"
-DminLeafSampleCount="1"
-DsampleRatio="1"
-DmaxDepth="10"
-DmetricType="0"
-DmodelName="gbdt_ls_test_model"
-DfeatureRatio="1"
-Dp="1"
-Dtau="0.6"
-DtestRatio="0"
-DfeatureColNames="f0,f1,f2,f3"
-DtreeCount="10"
```

- Prediction:

```
drop table if exists gbdt_ls_test_prediction_result;
PAI -name prediction
-project algo_public
-DdetailColName="prediction_detail"
-DmodelName="gbdt_ls_test_model"
-DitemDelimiter=","
-DresultColName="prediction_result"
-Dlifecycle="28"
-DoutputTableName="gbdt_ls_test_prediction_result"
-DscoreColName="prediction_score"
-DkvDelimiter=":"
-DinputTableName="gbdt_ls_test_input"
-DenableSparse="false"
-DappendColNames="label"
```

Input description

gbdt_ls_test_input

f0	f1	f2	f3	label
1.0	0.0	0.0	0.0	0
0.0	0.0	1.0	0.0	1
0.0	0.0	0.0	1.0	1
0.0	1.0	0.0	0.0	0
1.0	0.0	0.0	0.0	0
0.0	1.0	0.0	0.0	0

Output description

gbdt_ls_test_prediction_result

label	prediction_result	prediction_score	prediction_detail
0	NULL	0.0	{"label": 0}
0	NULL	0.0	{"label": 0}
1	NULL	0.9990234375	{"label": 0.9990234375}
1	NULL	0.9990234375	{"label": 0.9990234375}
0	NULL	0.0	{"label": 0}
0	NULL	0.0	{"label": 0}

4.4.6.4.2. Linear regression

This component is used to resolve regression issues and analyze the linear relationship between a dependent variable and multiple independent variables. Certain columns from an input table are selected as feature columns and one column is selected as the label column for linear regression training and linear regression model generation.

PAI command

```
PAI -name linearregression
  -project algo_public
  -DinputTableName=lm_test_input
  -DfeatureColNames=x
  -DlabelColName=y
  -DmodelName=lm_test_input_model_out;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
modelName	Required. The name of the output model.	-	-
outputTableName	Optional. The name of the output model evaluation table.	This parameter must be specified when enableFitGoodness is set to <i>true</i> .	-
labelColName	Required. The name of the label column.	The name must be a double or bigint type value. Only one column can be specified.	-

Parameter	Description	Valid values	Default value
featureColNames	Required. The name of the feature column.	The name must be a double or bigint type value in dense format, or a string type value in sparse format. Multiple columns can be specified.	-
inputTablePartitions	Optional. The partitions selected from the input table for training.	-	No partitions are selected by default.
maxIter	Optional. The maximum number of iterations.	-	100
epsilon	Optional. The minimum likelihood deviance.	-	0.000001
enableSparse	Optional. This parameter specifies whether the data is in sparse format.	true and false	false
enableFitGoodness	Optional. This parameter specifies whether to perform model evaluation. Model evaluation can be performed using a variety of metrics, including R-squared, adjusted R-Squared, Akaike information criterion, degrees of freedom, residual standard deviation, and deviation.	true and false	false
enableCoefficientEstimate	Optional. This parameter specifies whether to estimate the regression coefficient. The metrics of this parameter are value t, value p, and confidence interval [2.5%, 97.5%]. This parameter takes effect only when enableFitGoodness is set to <i>true</i> . This parameter is ignored when enableFitGoodness is set to <i>false</i> .	true and false	false
itemDelimiter	Optional. The delimiter used to separate key-value pairs. This parameter takes effect only when enableSparse is set to <i>true</i> .	-	Use spaces on command lines and use commas (,) on webpages.

Parameter	Description	Valid values	Default value
kvDelimiter	Optional. The delimiter used to separate keys and values. This parameter takes effect only when enableSparse is set to <i>true</i> .	-	The default delimiter is a colon (:).
lifecycle	Optional. The lifecycle of the output table.	An integer greater than or equal to -1	Default value: -1. This value indicates that no lifecycle is set.
coreNum	Optional. The number of cores.	An integer larger than 0	Default value: -1. This value indicates that the number of instances is determined by the amount of input data.
memSizePerCore	Optional. The memory size of each core.	(100, 65536)	Default value: -1. This value indicates that the memory size is determined by the amount of input data.

Examples

- SQL statement to generate data:

```
drop table if exists lm_test_input;
create table lm_test_input as
select
*
from
(
select 10 as y, 1.84 as x1, 1 as x2, '0:1.84 1:1' as sparsecol1 from dual
union all
select 20 as y, 2.13 as x1, 0 as x2, '0:2.13' as sparsecol1 from dual
union all
select 30 as y, 3.89 as x1, 0 as x2, '0:3.89' as sparsecol1 from dual
union all
select 40 as y, 4.19 as x1, 0 as x2, '0:4.19' as sparsecol1 from dual
union all
select 50 as y, 5.76 as x1, 0 as x2, '0:5.76' as sparsecol1 from dual
union all
select 60 as y, 6.68 as x1, 2 as x2, '0:6.68 1:2' as sparsecol1 from dual
union all
select 70 as y, 7.58 as x1, 0 as x2, '0:7.58' as sparsecol1 from dual
union all
select 80 as y, 8.01 as x1, 0 as x2, '0:8.01' as sparsecol1 from dual
union all
select 90 as y, 9.02 as x1, 3 as x2, '0:9.02 1:3' as sparsecol1 from dual
union all
select 100 as y, 10.56 as x1, 0 as x2, '0:10.56' as sparsecol1 from dual
) tmp;
```

- PAI command

```
PAI -name linearregression
  -project algo_public
  -DinputTableName=lm_test_input
  -DlabelColName=y
  -DfeatureColNames=x1,x2
  -DmodelName=lm_test_input_model_out
  -DoutputTableName=lm_test_input_conf_out
  -DenableCoefficientEstimate=true
  -DenableFitGoodness=true
  -Dlifecycle=1;
pai -name prediction
  -project algo_public
  -DmodelName=lm_test_input_model_out
  -DinputTableName=lm_test_input
  -DoutputTableName=lm_test_input_predict_out
  -DappendColNames=y;
```

- Output description:
 - When enableFitGoodness is set to *true*, partitions specified by `p='goodness'` are created in the model evaluation table. The output metrics are *R-squared*, *adjusted R-Squared*, *Akaike information criterion*, *degrees of freedom*, *residual standard deviation*, and *deviation*.
 - When enableCoefficientEstimate is set to *true*, partitions specified by `p='coefficient'` are created in the model evaluation table. The table contains the intercepts and the *name*, *coefficient*, *t-score*, *p-value*, and *confidence interval [2.5%, 97.5%]* of the features.

o Output model evaluation table: `lm_test_input_conf_out`.

colname	value	tscore	pvalue	confidenceinterval	p
Intercept	-6.42378496687763	-2.2725755951390028	0.06	{"2.5%": -11.964027, "97.5%": -0.883543}	coefficient
x1	10.260063429838898	23.270944360826963	0.0	{"2.5%": 9.395908, "97.5%": 11.124219}	coefficient
x2	0.35374498323846265	0.2949247320997519	0.81	{"2.5%": -1.997160, "97.5%": 2.704650}	coefficient
rsquared	0.9879675667384592	NULL	NULL	NULL	goodness
adjusted_rsquared	0.9845297286637332	NULL	NULL	NULL	goodness
aic	59.331109494251805	NULL	NULL	NULL	goodness
degree_of_freedom	7.0	NULL	NULL	NULL	goodness
standardErr_residual	3.765777749448906	NULL	NULL	NULL	goodness
deviance	99.26757440771128	NULL	NULL	NULL	goodness

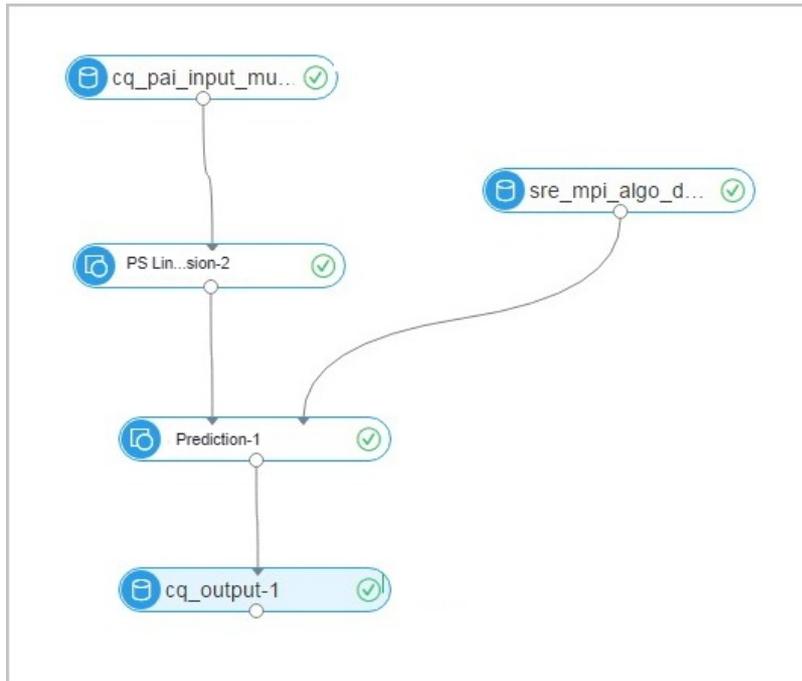
o Output prediction table: `lm_test_input_predict_out`.

y	prediction_result	prediction_score	prediction_detail
10	NULL	12.808476727264404	{"y": 12.8084767272644}
20	NULL	15.43015013867922	{"y": 15.43015013867922}
30	NULL	33.48786177519568	{"y": 33.48786177519568}
40	NULL	36.565880804147355	{"y": 36.56588080414735}
50	NULL	52.674180388994415	{"y": 52.67418038899442}
60	NULL	62.82092871092313	{"y": 62.82092871092313}
70	NULL	71.34749583130122	{"y": 71.34749583130122}
80	NULL	75.75932310613193	{"y": 75.75932310613193}
90	NULL	87.1832221199846	{"y": 87.18322211998461}
100	NULL	101.92248485222113	{"y": 101.9224848522211}

4.4.6.4.3. PS linear regression

Linear regression is a classic regression algorithm used to analyze the linear relationship between a dependent variable and multiple independent variables. Parameter servers (PSs) are used to run large amounts of training tasks online and offline. Parameter servers can use hundreds of billions of samples to efficiently train billions of feature models. The PS linear regression model can run training tasks with hundreds of billions of samples and billions of features, and supports L1 and L2 regular expressions.

Quick start



PAI command

- Training

```
PAI -name ps_linearregression  
-project algo_public  
-DinputTableName="lm_test_input"  
-DmodelName="linear_regression_model"  
-DlabelColName="label"  
-DfeatureColNames="features"  
-DL1Weight=1.0  
-DL2Weight=0.0  
-DmaxIter=100  
-Depsilon=1e-6  
-DenableSparse=true
```

- Prediction

```
drop table if exists logistic_regression_predict;
PAI -name prediction
-DmodelName="linear_regression_model"
-DoutputTableName="linear_regression_predict"
-DinputTableName="lm_test_input"
-DdependColNames="label,features"
-DfeatureColNames="features"
-DenableSparse=true
```

Parameters

- Data parameters

Command option	Parameter	Description	Valid values	Default value
featureColNames	Feature Columns	Required. The names of feature columns selected from the input table for training.	If a column name is in dense format, it must be of the bigint or double type. If the column name is in sparse KV format, it must be a string.	-
labelColName	Label Column	Required. The name of the label column selected from the input table.	The column name must be of the bigint or double type.	-
enableSparse	Use Sparse Format	Optional. If you choose to use the sparse KV format, do not use feature ID 0. We recommend that the feature IDs start from 1.	true and false	false
itemDelimiter	KV Pair Delimiter	Optional. The delimiter used to separate key-value pairs when data in the input table is in sparse format.	Symbol	The default delimiter is a space.
kvDelimiter	KV Delimiter	Optional. The delimiter used to separate keys and values when data in the input table is in sparse format.	Symbol	The default delimiter is a colon (:).
inputTableName	Input Table Name	Required.	Table name	-
modelName	Output Model Name	Required.	Model name	-

Command option	Parameter	Description	Valid values	Default value
inputTablePartitions	Input Table Partitions	Optional.	Partition name	The parameter value must be in the ds=1/pt=1 format.
enableModello	Output to Offline Model	Optional. When this parameter is set to false, the data is output to a MaxCompute table where you can view model weights.	true and false	true

• **Algorithm parameters**

Command option	Parameter	Description	Valid values	Default value
l1Weight	L1 Weight	Optional. The L1 regularization coefficient. The larger this value is, the fewer non-zero elements a model has. To overfit the model, set this parameter to a larger value.	A non-negative real number	1.0
l2Weight	L2 Weight	Optional. The L2 regularization coefficient. The larger this value is, the smaller the absolute values of the model parameters are. To overfit the model, set this parameter to a larger value.	A non-negative real number	0
maxIter	Maximum Iterations	Optional. The maximum number of LBFGS/OWL-QN iterations. Value 0 indicates that no limit is set.	A non-negative integer	100

Command option	Parameter	Description	Valid values	Default value
epsilon	Minimum Convergence Deviance	Optional. The mean of the relative loss change rates in ten iterations, which is used as a condition to determine whether to terminate the optimization algorithm. The smaller this value is, the stricter the condition is, and the longer the algorithm runs.	A real number between 0 and 1	1.0e-06
modelSize	Largest Feature ID	Optional. The largest feature ID among all feature IDs (feature dimension). It can be larger than the actual largest feature ID. The larger this value is, the higher the memory usage is. If you leave this parameter empty, the system starts an SQL task to calculate the largest feature ID automatically.	A non-negative integer	0

 **Note** Both the maximum iterations and minimum convergence deviance determine when the algorithm stops. If both parameters are set, the algorithm stops when one of the conditions is met.

- **Execution optimization**

Command option	Parameter	Description	Valid values	Default value
coreNum	Cores	Optional. The number of cores. The larger this value is, the faster the computing algorithm runs.	A positive integer	Automatically allocated.

Command option	Parameter	Description	Valid values	Default value
<code>memSizePerCore</code>	Memory Size per Core (MB)	Optional. The memory size of each core, where 1024 represents 1 GB of memory.	A positive integer	Automatically allocated. Typically, you do not need to set this parameter because the algorithm can accurately estimate the memory size required.

Examples

- **Data generation**

The following example uses data in sparse KV format:

```
drop table if exists lm_test_input;
create table lm_test_input as
select
*
from
(
select 2 as label, '1:0.55 2:-0.15 3:0.82 4:-0.99 5:0.17' as features from dual
union all
select 1 as label, '1:-1.26 2:1.36 3:-0.13 4:-2.82 5:-0.41' as features from dual
union all
select 1 as label, '1:-0.77 2:0.91 3:-0.23 4:-4.46 5:0.91' as features from dual
union all
select 2 as label, '1:0.86 2:-0.22 3:-0.46 4:0.08 5:-0.60' as features from dual
union all
select 1 as label, '1:-0.76 2:0.89 3:1.02 4:-0.78 5:-0.86' as features from dual
union all
select 1 as label, '1:2.22 2:-0.46 3:0.49 4:0.31 5:-1.84' as features from dual
union all
select 0 as label, '1:-1.21 2:0.09 3:0.23 4:2.04 5:0.30' as features from dual
union all
select 1 as label, '1:2.17 2:-0.45 3:-1.22 4:-0.48 5:-1.41' as features from dual
union all
select 0 as label, '1:-0.40 2:0.63 3:0.56 4:0.74 5:-1.44' as features from dual
union all
select 1 as label, '1:0.17 2:0.49 3:-1.50 4:-2.20 5:-0.35' as features from dual
) tmp;
```

The feature IDs start from 1, and the maximum feature ID is 5.

- **Training**

Configure the training data and training components based on [Quick start](#). Select the label column as the target column and features column as the feature column. Then, select the sparse data format.

- You can retain the default value 0 for the largest feature ID. The algorithm can start an SQL task to calculate the largest feature ID automatically. If you do not want to start the SQL task, enter a value greater than 5. This value indicates the number of feature columns in dense format and indicates the largest feature ID in KV format.

- To accelerate the training, you can set the number of cores on the tuning page. The larger the number is, the faster the algorithm runs. Typically, you do not need to enter the memory size per core because the algorithm can accurately calculate the memory size. The PS algorithm starts to run only when all hosts have obtained the resources. Therefore, you may need to wait a longer period of time when the cluster is busy and resources are requested in large volume.

- **Prediction**

The model generated after training is saved in binary format and can be used for prediction. Configure the input settings (model and testing data) for the prediction component and set parameters based on [Quick start](#).

Select the KV format for training and set a correct delimiter. When the KV format is used, key-value pairs are separated by spaces. Therefore, the delimiter must be set to space or `\u0020` (the escape expression of space).

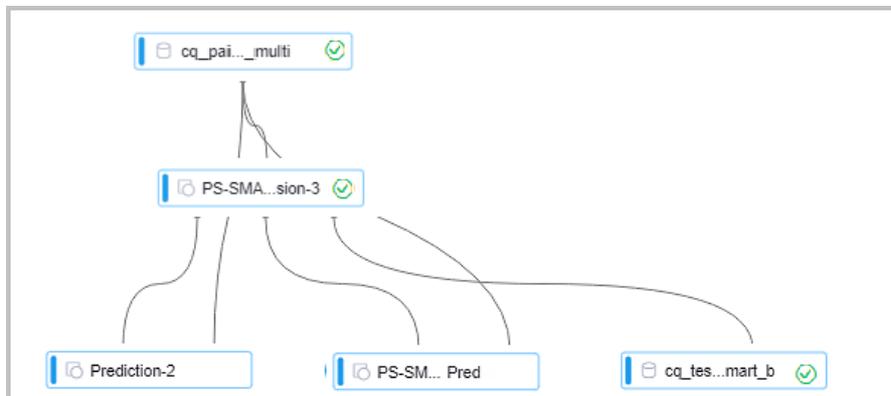
Restrictions and guidelines

In the key-value format, feature IDs must be positive integers, and feature values must be real numbers. If feature IDs are strings, use the serialization component to serialize them. If the feature values are classification type strings, perform feature engineering, such as discretization.

4.4.6.4.4. PS-SMART regression

A [parameter server](#) (PS) is used to train a large number of models online and offline. Scalable Multiple Additive Regression Tree (SMART) is an implementation of Gradient boosting decision tree (GBDT) on PS. PS-SMART can run training tasks containing up to tens of billions of samples and hundreds of thousands of features on thousands of nodes. It also supports failover for high stability. PS-SMART supports various data formats, training targets, evaluation targets, output feature importance, and histogram approximation for training acceleration.

Quick start



As shown in the figure, a PS-SMART regression model is learned based on training data. The model has three output ports:

- Output model: offline model, which is connected to the unified prediction component. This model does not support the output of leaf node numbers.
- Output model table: a binary table that is not readable and is used to ensure compatibility with the PS-SMART prediction component. The table provides outputs such as leaf node numbers and evaluation metrics. However, the output table has strict requirements on data formats, which negatively affects user experience. This component is being continually improved, and may be replaced by another component in the future.
- Output feature importance table: lists the importance of each feature. Three importance types are supported. For more information, see [Parameters](#).

PAI command

- Training

```
PAI -name ps_smart
  -project algo_public
  -DinputTableName="smart_regression_input"
  -DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
  -DoutputTableName="pai_temp_24515_545859_2"
  -DoutputImportanceTableName="pai_temp_24515_545859_3"
  -DlabelColName="label"
  -DfeatureColNames="features"
  -DenableSparse="true"
  -Dobjective="reg:linear"
  -Dmetric="rmse"
  -DfeatureImportanceType="gain"
  -DtreeCount="5";
  -DmaxDepth="5"
  -Dshrinkage="0.3"
  -DI2="1.0"
  -DI1="0"
  -Dlifecycle="3"
  -DsketchEps="0.03"
  -DsampleRatio="1.0"
  -DfeatureRatio="1.0"
  -DbaseScore="0.5"
  -DminSplitLoss="0"
```

- Prediction

```
PAI -name prediction
  -project algo_public
  -DinputTableName="smart_regression_input";
  -DmodelName="xlab_m_pai_ps_smart_bi_545859_v0"
  -DoutputTableName="pai_temp_24515_545860_1"
  -DfeatureColNames="features"
  -DappendColNames="label,features"
  -DenableSparse="true"
  -Dlifecycle="28"
```

Parameters

- Data parameters

Command option	Parameter	Description	Valid values	Remarks
featureColNames	Feature Column	The names of feature columns selected from the input table for training.	If the column name is in dense format, it must be of the bigint or double type. If the column name is in sparse KV format, it must be a string, and its keys and values must be numeric.	Required

Command option	Parameter	Description	Valid values	Remarks
labelColName	Label Column	The name of the label column selected from the input table.	The column name can be of either string or numeric type, but only numeric data can be stored in the columns. For example, the column value can be 0 or 1 for regression.	Required
weightCol	Weight Column	This column specifies the weight of each sample.	The column name can be of the numeric type.	Optional. Default value: null.
enableSparse	Use Sparse Format	This parameter specifies whether the data in the input table is in sparse format, in which key-value pairs are separated by spaces whereas keys and values are separated by colons (:), for example, 1:0.3 3:0.9 .	true, false	Optional. Default value: false.
inputTableName	Input Table Name	N/A	N/A	Required
modelName	Output Model Name	N/A	N/A	Required
outputImportanceTableName	Output Feature Importance Table Name	N/A	N/A	Optional. Default value: null.
inputTablePartitions	Input Table Partitions	N/A	N/A	Optional. The parameter value must be in ds=1/pt=1 format.
outputTableName	Output Model Table Name	The output table is a MaxCompute table that uses the binary format and is not readable. The prediction component that comes with SMART can be used to generate leaf node numbers.	String	Optional

Command option	Parameter	Description	Valid values	Remarks
lifecycle	Output Table Lifecycle	N/A	Positive integer	Optional. Default value: 3.

- Algorithm parameters

Command option	Parameter	Description	Valid values	Remarks
objective	Objective Function Type	The objective function type affects learning and must be selected properly. Multiple loss functions are available for regression. For more information, see the notes.		Required. The default type is Linear regression.
metric	Evaluation Indicator Type	Evaluation indicators in the training set, which must correspond to the objective function type and are exported to stdout of the coordinator in a logview. For more information, see the following notes and samples.		Optional. Default value: null.
treeCount	Trees	The number of trees. The training time is proportional to this number.	Positive integer	Optional. Default value: 1.
maxDepth	Maximum Decision Tree Depth	The maximum depth of a tree. We recommend that you set this value to 5, which means the tree can contain up to 32 leaf nodes.	A positive integer in the range of [1, 20]	Optional. Default value: 5.
sampleRatio	Data Sampling Fraction	The data sampling rate when trees are built. The sample data is used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means data sampling is disabled.

Command option	Parameter	Description	Valid values	Remarks
featureRatio	Feature Sampling Fraction	The feature sampling rate when trees are built. The sample features are used to build a weak learner to accelerate training.	(0, 1]	Optional. The default value is 1.0, which means feature sampling is disabled.
l1	L1 Penalty Coefficient	This parameter determines the number of leaf nodes. The greater the value, the fewer the leaf nodes. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 0.
l2	L2 Penalty Coefficient	This parameter determines the size of a leaf node. The greater the value, the more evenly the leaf nodes are distributed. You can set this parameter to a greater value if overfitting occurs.	Non-negative real number	Optional. Default value: 1.0.
shrinkage	Learning Rate	N/A	(0, 1]	Optional. Default value: 0.3.
sketchEps	Sketch-based Approximate Precision	The threshold for selecting quantiles when you build a sketch. The number of buckets is $O(1.0/sketchEps)$. The smaller the parameter value, the more buckets are generated. Typically, you do not need to change this value.	(0, 1)	Optional. Default value: 0.03.

Command option	Parameter	Description	Valid values	Remarks
minSplitLoss	Minimum Split Loss	The minimum split loss changes required for splitting a node. The greater the value, the more conservatively the node splits.	Non-negative real number	Optional. Default value: 0.
featureNum	Features	The number of features or the maximum feature ID. Specify this parameter for resource usage estimation.	Positive integer	Optional
baseScore	Global Offset	Original predicted values of all samples.	Real number	Optional. Default value: 0.5.
featureImportance Type	Feature Importance Type	The type of feature importance. weight indicates the number of times that a feature splits. gain indicates information gain brought by the feature. cover indicates the number of samples that the feature covers on the splitting nodes.	weight , gain , and cover	Optional. Default value: gain .
tweedieVarPower	Tweedie Distribution Index	Tweedie distribution index indicates the relationship between the variance and mean. For example, $\text{Var}(y) \sim E(y)^{\text{tweedie_variance_power}}$.	(1, 2)	Optional. Default value: 1.5.

- Note

- Specify different values for the objective parameter in different learning models. The regression Web GUI provides multiple objective functions.

reg:linear (Linear regression) // The range of label numbers is $(-\infty, +\infty)$.
 reg:logistic (Logistic regression) // The range of label numbers is $[0, 1]$.
 count:poisson (Poisson regression for count data, output mean of poisson distribution) // Label numbers must be greater than 0.
 reg:gamma (Gamma regression for modeling insurance claims severity, or for any outcome that might be [gamma-distributed](https://en.wikipedia.org/wiki/Gamma_distribution#Applications)) // Label numbers must be greater than 0.
 reg:tweedie (Tweedie regression for modeling total loss in insurance, or for any outcome that might be [Tweedie-distributed](https://en.wikipedia.org/wiki/Tweedie_distribution#Applications).) // Label numbers must be greater than or equal to 0.

- Metrics for these objective functions are:

rmse (rooted mean square error, corresponding to objective reg:linear)
 mae (mean absolute error, corresponding to objective reg:linear)
 poisson-nloglik (negative loglikelihood for poisson regression, corresponding to objective count:poisson)
 gamma-deviance (Residual deviance for gamma regression, corresponding to objective reg:gamma)
 gamma-nloglik (Negative log-likelihood for gamma regression, corresponding to objective reg:gamma)
 tweedie-nloglik (tweedie-nloglik@1.5, negative log-likelihood for Tweedie regression, at a specified value of the tweedie_variance_power parameter)

- Execution optimization

Command option	Parameter	Description	Valid values	Remarks
coreNum	Cores	The number of cores. The greater the value, the faster the computing algorithm runs.	Positive integer	Optional. Automatically allocated.
memSizePerCore	Memory Size per Core (MB)	The memory size of each core, where 1024 represents 1 GB of memory.	Positive integer	Optional. Automatically allocated.

Example

- Data generation

The following example uses data in sparse KV format.

```
drop table if exists smart_regression_input;
create table smart_regression_input as
select
*
from
(
select 2.0 as label, '1:0.55 2:-0.15 3:0.82 4:-0.99 5:0.17' as features from dual
union all
select 1.0 as label, '1:-1.26 2:1.36 3:-0.13 4:-2.82 5:-0.41' as features from dual
union all
select 1.0 as label, '1:-0.77 2:0.91 3:-0.23 4:-4.46 5:0.91' as features from dual
union all
select 2.0 as label, '1:0.86 2:-0.22 3:-0.46 4:0.08 5:-0.60' as features from dual
union all
select 1.0 as label, '1:-0.76 2:0.89 3:1.02 4:-0.78 5:-0.86' as features from dual
union all
select 1.0 as label, '1:2.22 2:-0.46 3:0.49 4:0.31 5:-1.84' as features from dual
union all
select 0.0 as label, '1:-1.21 2:0.09 3:0.23 4:2.04 5:0.30' as features from dual
union all
select 1.0 as label, '1:2.17 2:-0.45 3:-1.22 4:-0.48 5:-1.41' as features from dual
union all
select 0.0 as label, '1:-0.40 2:0.63 3:0.56 4:0.74 5:-1.44' as features from dual
union all
select 1.0 as label, '1:0.17 2:0.49 3:-1.50 4:-2.20 5:-0.35' as features from dual
) tmp;
```

Feature IDs are numbered starting from 1, and the maximum feature ID is 5.

- Training

Select the label column as the target column and the features column as the feature column.

- You do not need to set the number of features because this number is calculated automatically by the algorithm. If you have a large number of features and want the algorithm to accurately estimate the amount of required resources, specify the actual number of features.
- To accelerate the training, set the number of cores on the execution optimization page. The greater the number, the faster the algorithm runs. Typically, you do not need to enter the memory size per core because the algorithm can accurately calculate the memory size. The PS algorithm starts to run only when all hosts have obtained the required resources. Therefore, you may need to wait for a longer time when the cluster is busy and resources are requested in large volumes.
- You can view the output values of the metrics in the stdout of the coordinator in a logview (HTTP link starting with <http://logview.odps.aliyun-inc.com:8080/logview>). A single PS-SMART training job can contain multiple tasks, and therefore multiple logviews are created. Select the logview whose name starts with PS to view the output of the PS job.

- Prediction

- Use the unified prediction component

The model generated after training is saved in binary format and can be used for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#).

If the dense format is used, you only need to select feature columns. (All columns are selected by default, and extra columns do not affect the prediction.) If the KV format is used, set the data format to sparse format and select the correct delimiter. In the SMART model, key-value pairs are separated with spaces. Therefore, the delimiter must be set to space or `\u0020` (escape expression of spaces).

- Use the PS-SMART prediction component

The output model table obtained after training is saved in binary format and can be used by the PS-SMART prediction component for prediction. Configure the input model and test data for the prediction component, as shown in [Quick start](#). Set the required parameters, including the data format, feature columns, target column, and number of classes. The ID column can only be a string type column other than a feature column or a target column. The loss function must be explicitly set to the objective function used for training.

The **prediction_score** column lists the predicted values. The **leaf_index** column lists the predicted leaf node numbers. Each sample has N numbers, where N is the number of decision trees. Each tree is mapped to a number, which indicates the leaf node number of the sample on this tree.

 Note

- The output model table is a binary table that is not readable and is used to support the PS-SMART prediction component. The table provides outputs such as leaf node numbers and evaluation indicators. However, the output table has strict requirements on data formats, which negatively affects user experience. This component is being continually improved, and may be replaced by another component in the future.
- A string type column must be selected as the label column. You can enter strings in the column but cannot be blank or NULL. A feature column can be converted to the string type by using the data type conversion component.
- The loss function must be explicitly set to the objective function used for training. By default, the loss function does not work.

- View feature importance

To view feature importance, you can export the third output port to an output table, or right-click **PS-SMART training component** and choose **View Data > Output Feature Importance Table** from the shortcut menu.

order ▲	id ▲	value ▲
1	1	0.14059734344482422
2	4	0.8594027161598206

In the table, the ID column lists the numbers of input features. In this example, the data is in KV format and the IDs represent keys in key-value pairs. If the dense format is used and input features are **f0,f1,f2,f3,f4,f5**, ID 0 represents f0, and ID 4 represents f4. Each value indicates a feature importance type. The default value is **gain**, indicating the sum of information gains brought by a feature in the model. The preceding figure shows only two features because only these two features are used during the tree split process. In this case, the importance of unused features is 0.

FAQ

- Q: Does PS_SMART support non-numerical features and tags?
- A: No.
- Q: What is the scale of features supported by PS-SMART? Can we use large-scale 0-1 features?
- A: Although PS-SMART supports tasks that contain hundreds of thousands of features, such tasks consume large amounts of resources and run slowly. Therefore, we recommend that you do not use a large number of features. The GBDT algorithm is suitable for training with continuous features. The categorical features require one-hot coding (to filter out infrequent features) before they can be used for training. The continuous numerical features can be used for training with the GBDT algorithm directly. Discretization is not recommended for numerical features.
- Q: Why is the result different every time although the SMART algorithm has the same data and the same

parameter settings?

- A: The PS-SMART algorithm applies randomness in many scenarios. For example, the `data_sample_ratio` and `fea_sample_ratio` items introduce data and feature sampling respectively. In addition, the PS-SMART algorithm uses histograms to show similarity. When multiple workers run in a cluster in distributed mode, local sketches are merged to global sketches in a random order. Although different merging orders result in different tree structures, this does not introduce too much variation to the output model. Therefore, it is normal situation to obtain different results after the algorithm runs multiple times with the same data and same parameter settings.

Note

- The target column in a PS-SMART regression model supports only numerical values. Even if values in the MaxCompute table are strings, they are saved as numerical values.
- In the key-value format, feature IDs must be positive integers, and feature values must be real numbers. If feature IDs are strings, use the serialization component to serialize them. If feature values are classification type strings, perform feature engineering, such as discretization.

4.4.6.5. Collaborative filtering (etrec)

etrec is an item-based collaborative filtering algorithm that takes two input columns and provides the top N items that have the highest similarity.

Set the user and item columns.

- You can configure three similarity types.
- topN indicates the first N items with the highest similarity.
- Calculation method: the method used to calculate items that appear multiple times.

PAI command

```
PAI -name pai_etrec
-project algo_public
-DsimilarityType="wbcosine"
-Dweight="1"
-DminUserBehavior="2"
-Dlifecycle="28"
-DtopN="2000"
-Dalpha="0.5"
-DoutputTableName="etrec_test_result"
-DmaxUserBehavior="500"
-DinputTableName="etrec_test_input"
-Doperator="add"
-DuserColName="user"
-DitemColName="item"
```

Parameters

Parameters

Parameter	Description	Valid value	Default value
<code>inputTableName</code>	Required. The name of the input table.	N/A	N/A

Parameter	Description	Valid value	Default value
userColName	Required. The name of the input table column selected as the user column.	N/A	N/A
itemColName	The name of the input table column selected as the item column.	N/A	N/A
payloadColName	Optional. The name of the input table column selected as the payload column.	N/A	No payload column is set by default.
inputTablePartitions	Optional. The partitions selected from the input table for training.	N/A	The whole table is selected by default.
outputTableName	Required. The name of the output table.	N/A	N/A
outputTablePartition	Optional. The partitions in the output table.	N/A	The output table is non-partitioned by default.
similarityType	Optional. The type of similarity.	wbcosine, asymcosine, and jaccard	wbcosine
topN	Optional. N items with the highest similarity.	[1, 10000]	2000
minUserBehavior	Optional. The minimum user behavior.	[2,)	2
maxUserBehavior	Optional. The maximum user behavior.	[2, 100000]	500
itemDelimiter	Optional. The delimiter used to separate items in the output table.	N/A	The default delimiter is a space.
kvDelimiter	Optional. The delimiter used to separate keys and values in the output table.	N/A	The default delimiter is a colon (:).
alpha	Optional. The value of the smoothing factor for asymcosine.	N/A	0.5
weight	Optional. The weight used for asymcosine.	N/A	1.0
operator	Optional. The action to be performed when the same items exist for one user.	add, mul, min, and max	add

Parameter	Description	Valid value	Default value
lifecycle	Optional. The lifecycle of the output table.	N/A	1

Examples

- SQL statement to generate data:

```
drop table if exists etrec_test_input;
create table etrec_test_input as select * from
(
select cast(0 as string) as user,
cast(0 as string) as item from dual
union all
select cast(0 as string) as user,
cast(1 as string) as item from dual
union all
select cast(1 as string) as user,
cast(0 as string) as item from dual
union all
select cast(1 as string) as user,
cast(1 as string) as item from dual ) a;
```

- PAI command

```
drop table if exists etrec_test_result;
PAI -name pai_etrec
-project algo_public
-DsimilarityType="wbcosine"
-Dweight="1"
-DminUserBehavior="2"
-Dlifecycle="28"
-DtopN="2000"
-Dalpha="0.5"
-DoutputTableName="etrec_test_result"
-DmaxUserBehavior="500"
-DinputTableName="etrec_test_input"
-Doperator="add"
-DuserColName="user"
-DitemColName="item"
```

- Input description etrec_test_input

User	Item
0	0
0	1
1	0
1	1

- Output description etrec_test_result

Item ID	Similarity
0	1:1
1	0:1

4.4.6.6. Evaluation

4.4.6.6.1. Regression model evaluation

You can evaluate a regression model based on the predicted and actual results. Indicators include SST, SSE, SSR, R2, R, MSE, RMSE, MAE, MAD, MAPE, count, yMean, and predictMean.

PAI command

```
Pai -name regression_evaluation
    -project algo_public
    -DinputTableName=input_table
    -DyColName=y_col
    -DpredictionColName=prediction_col
    -DoutputTableName=output_table;
```

Parameters

Parameter	Description	Default value
inputTableName	Required. The name of the input table.	-
inputTablePartitions	Optional. The partitions selected from the input table for training.	All partitions of the input table are selected by default.
yColName	Required. The name of the original dependent variable column in the input table. It must be a numerical value.	-
predictionColName	Required. The name of the predicted dependent variable column. It must be a numerical value.	-
outputTableName	Required. The name of the output table.	-
inputTablePartitions	Optional. The partitions selected from the input table.	-
lifecycle	Optional. The lifecycle of the output table.	No lifecycle is set by default.

Output

The following table describes the JSON columns.

Column description

Column	Description
SST	The sum of squares total.

Column	Description
SSE	The sum of squares error.
SSR	The sum of squares regression.
R2	The coefficient of determination.
R	The coefficient of multiple correlations.
MSE	The mean squared error.
RMSE	The root-mean-square error.
MAE	The mean absolute error.
MAD	The mean absolute difference.
MAPE	The mean absolute percentage error.
count	The number of rows.
yMean	The mean of original dependent variables.
predictionMean	The mean of prediction results.

4.4.6.6.2. Clustering model evaluation

You can evaluate clustering models, including metrics and icons, based on raw data and clustering models.

PAI command

```
PAI -name cluster_evaluation
    -project algo_public
    -DinputTableName=pai_cluster_evaluation_test_input
    -DselectedColNames=f0,f3
    -DmodelName=pai_kmeans_test_model
    -DoutputTableName=pai_ft_cluster_evaluation_out;
```

Parameters

Parameters

Parameter	Description	Valid value	Default value
inputTableName	Required. The name of the input table.	Table name	N/A

Parameter	Description	Valid value	Default value
selectedColNames	Optional. The names of columns selected from the input table for evaluation. The column names must be separated with commas (.). The column names must be the same as the feature names saved in the model.	Column name	All columns are selected by default.
inputTablePartitions	Optional. The partitions selected from the input table for evaluation, in the <code>name1=value1/name2=value2</code> format. Separate multiple partitions with commas (.).	N/A	All partitions are selected by default.
enableSparse	Optional. This parameter specifies whether data in the input table is in sparse format.	true and false	false
itemDelimiter	Optional. The delimiter used to separate key-value pairs when data in the input table is in sparse format.	N/A	The default delimiter is a space.
kvDelimiter	Optional. The delimiter used to separate keys and values when data in the input table is in sparse format.	N/A	The default delimiter is a colon (:).
modelName	Required. The name of the input clustering model.	Model name	N/A
outputTableName	Required. The name of the output table.	Table name	N/A
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Evaluation formula

The Calinski-Harabasz metric is also known as the variance ratio criterion (VRC), which is defined as follows:

$$VRC_k = \frac{SS_B}{SS_W} \times \frac{(N-k)}{(k-1)},$$

- SS_B represents the inter-clustering variance. The definition is as follows:

$$SS_B = \sum_{i=1}^k n_i \|m_i - m\|^2,$$

- **k** represents the number of cluster centers.
- m_i represents the center of cluster i .
- **m** represents the mean of the input data.
- SS_W represents the intra-clustering variance. The definition is as follows:

$$SS_W = \sum_{i=1}^k \sum_{x \in C_i} \|x - m_i\|^2,$$

- **k** represents the number of cluster centers.
- x represents a data point.
- C_i represents the number i cluster.
- m_i represents the center of cluster i .
- **N** represents the total number of records. **k** represents the number of cluster centers.

Examples

- Test data

```
create table if not exists pai_cluster_evaluation_test_input
as select * from ( select 1 as id,
1 as f0,2 as f3 from dual union all
select 2 as id, 1 as f0,3 as f3 from dual union all
select 3 as id, 1 as f0,4 as f3 from dual union all
select 4 as id, 0 as f0,3 as f3 from dual union all
select 5 as id, 0 as f0,4 as f3 from dual )tmp;
```

- Clustering model building

```
pai -name kmeans
-project algo_public
-DinputTableName=pai_cluster_evaluation_test_input
-DselectedColNames=f0,f3
-DcenterCount=3
-Dloop=10
-Daccuracy=0.00001
-DdistanceType=euclidean
-DinitCenterMethod=random
-Dseed=1
-DmodelName=pai_kmeans_test_model
-DidxTableName=pai_kmeans_test_idx
```

- PAI command

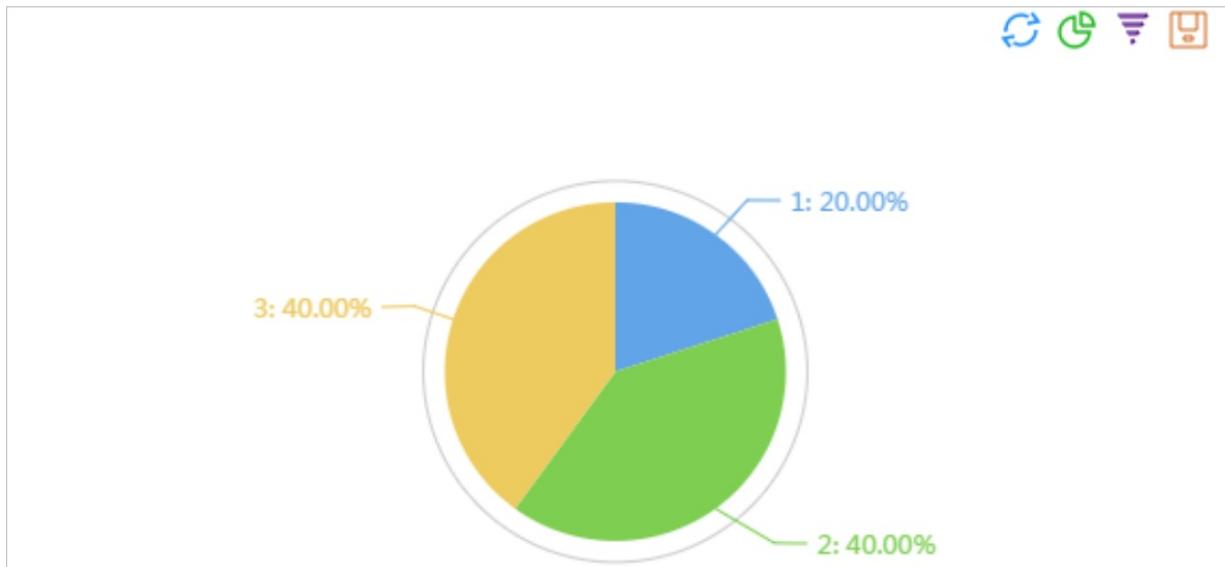
```
PAI -name cluster_evaluation
-project algo_public
-DinputTableName=pai_cluster_evaluation_test_input
-DselectedColNames=f0,f3
-DmodelName=pai_kmeans_test_model
-DoutputTableName=pai_ft_cluster_evaluation_out;
```

- Output description Output table (outputTableName)

Column	Description
count	The total number of records.
centerCount	The number of cluster centers.
calinhara	The Calinski Harabasz metric.
clusterCounts	The number of points included in each cluster.

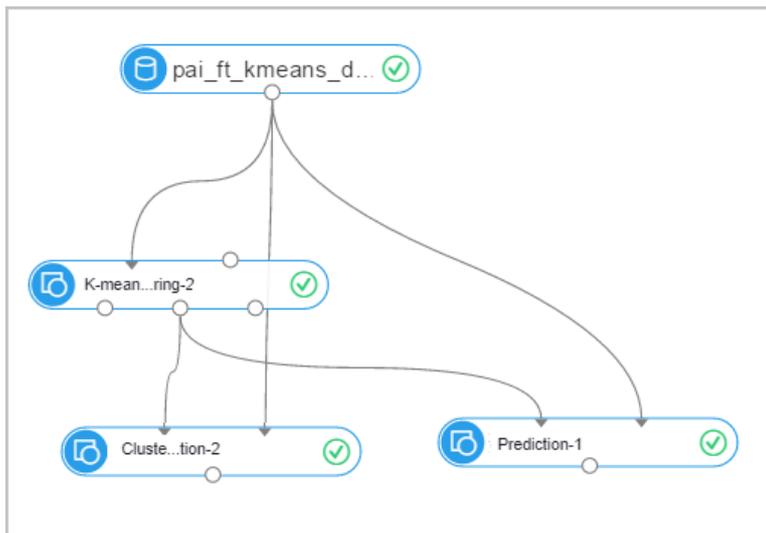
PaiWeb demonstration

Clustering model evaluation



PaiWeb-Pipeline

PaiWeb-Pipeline



4.4.6.6.3. Binary classification evaluation

You can evaluate a regression algorithm model based on its predicted and actual results. The metrics include MSE, MAE, and MAPE.

PAI command

```
pai -name evaluation
-DinputTableName=input_table
-DlabelColName=label_name
-DpredictionColName=prediction_score
-DoutputTableName=output_table;
```

Algorithm parameters

Parameters

Parameter	Description	Valid value	Default value
inputTableName	Required. The name of the input table.	N/A	N/A
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	N/A	All partitions of the input table are selected by default.
labelColName	Required. The name of the original label column in the input table. It must be a numerical value.	N/A	N/A
predictionColName	Required. The name of the label column in the prediction result table. It must be a numerical value.	N/A	N/A
outputTableName	Required. The name of the output table.	N/A	N/A
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Output table

Output column description

Column	Description
MSE	The mean square error, which is used to measure the mean error and evaluate data changes. The smaller the MSE, the more accurately a prediction model describes the test data.
MAE	The mean absolute error, which is used to measure the mean difference between the predicted value and the actual value.
MAPE	The mean absolute percentage error, which is used to measure prediction accuracy. The value is expressed in percentage. If MAPE is set to 15, the mean absolute percentage error is 15%.

4.4.6.6.4. Confusion matrix

The Confusion Matrix component is a visualization tool typically used in supervised learning. This tool is used to calculate the classification accuracy of a confusion matrix model by comparing its results with measured values.

Procedure

1. Configure the confusion matrix parameters.

The default settings are typically used. You can also select a target column and a prediction probability column. A prediction probability column is the target column generated by the Prediction component.

2. Connect the Confusion Matrix component and the Prediction component.

 **Note** The parent node of the Confusion Matrix component must be a Prediction component. You can perform confusion matrix analysis only when a classification model is used.

3. Right-click the Confusion Matrix component and choose **View Evaluation Report**.

PAI command

```
PAI -name confusionmatrix  
-project algo_public  
-DoutputTableName="pai_temp_2954_24178_1"  
-DlabelColName="age"  
-DpredictionColName="prediction_result"  
-DinputTableName="pai_temp_2954_24176_1";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is algo_public. If you change the name, the system reports an error.
outputTableName	The name of the output table.
labelColName	The name of the label column selected from the input table.
predictionColName	The name of the prediction result column.
inputTableName	The name of the input table for predicting results.

4.4.6.6.5. Multiclass classification evaluation

You can evaluate a multiclass classification model based on its predicted and actual results. The indicators include accuracy, kappa, and F1-Score.

Component description

The Multiclass Classification Evaluation component must be connected to a Prediction component and does not support regression models.

PAI command

```
PAI -name MultiClassEvaluation -project algo_public
-DinputTableName="test_input"
-DoutputTableName="test_output"
-DlabelColName="label"
-DpredictionColName="prediction_result"
-Dlifecycle=30;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for training.	-	All partitions of the input table are selected by default.
labelColName	Required. The name of the original label column in the input table. It must be a numerical value.	-	-
predictionColName	Required. The name of the label column in the prediction result table. It must be a numerical value.	-	-
outputTableName	Required. The name of the output table.	-	-
predictionColName	Optional. The name of the probability column that lists prediction results. It must be in the {"A":0.2,"B":0.3} format.	-	-
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

4.4.6.7. Prediction

The Prediction component is used to make model-based predictions. The component has two inputs (training model and prediction data) and one output (prediction result). Conventional data mining algorithms often use this component for prediction.

Procedure

1. Connect all components.
2. Configure column settings. Parameters

Parameter	Description
Feature Columns	The feature columns used for prediction. All feature columns are selected by default.
Reserved Columns	The columns reserved and exported to the prediction result.

Parameter	Description
Output Result Column	The default value is used.
Output Score Column	The default value is used.
Output Detail Column	The default value is used.

 **Note** Feature columns must be selected if data is in sparse format such as `k1:v1,k2:v2` .

- After you configure the preceding parameters, right-click the Prediction component and choose **View Data** from the short cut menu.

The following three columns are appended to the prediction data:

- `predict_result`: the prediction result column.
- `predict_score`: the probability score in prediction results. This column is only appended onto the outputs of binary classification models.
- `predict_detail`: the prediction result of each category. This column is only appended onto the outputs of binary classification models.

PAI command

```
PAI -name Prediction
-project algo_public
-DdetailColName="prediction_detail"
-DsplitCharacteristic="2"
-DappendColNames="age,campaign,pdays,previous,poutcome,emp_var_rate,cons_price_idx,cons_conf_idx,euribor3m,nr_employed,y"
-DmodelName="xlab_m_random_forests_6036"
-DresultColName="prediction_result"
-DoutputTableName="pai_temp_675_6048_1"
-DscoreColName="prediction_score"
-DinputTableName="bank_data";
```

Parameters

Parameter	Description
name	The name of the component.
project	The name of the project. This parameter is used to specify the workspace of the algorithm. The default value is <code>algo_public</code> . If you change the name, the system reports an error.
detailColName	Optional. The name of the detail column in the output table. The default value is <code>prediction_detail</code> .
splitCharacteristic	Optional. The type of classification. The value <code>1</code> indicates binary classification. The value <code>2</code> indicates multiclass classification.
appendColNames	Optional. The names of columns in the input prediction table to be appended to the output table.
modelName	The name of the random forest model.

Parameter	Description
<code>resultColName</code>	Optional. The name of the result column in the output table. The default value is <i>prediction_result</i> .
<code>outputTableName</code>	The name of the output prediction table.
<code>scoreColName</code>	Optional. The name of the score column in the output table. The default value is <i>prediction_score</i> .
<code>inputTableName</code>	The name of the input prediction table.

4.4.7. Deep learning (must be activated separately)

4.4.7.1. Activate deep learning

The deep learning service is not a basic function of Apsara Stack Machine Learning Platform for AI. You must purchase it separately.

If you have already deployed the deep learning service, activate it by using the following procedure:

1. Log on to the Apsara Stack Machine Learning Platform for AI console.
2. Click **Settings** in the left-side navigation pane.
3. Click **General**. Under **Deep Learning**, select **Enable GPU Compute**.

4.4.7.2. Read OSS buckets

When using the **Read OSS Bucket** component on Machine Learning Platform for AI, you must assign the default system role **AliyunODPSAIDefaultRole** to your DTplus service account. OSS buckets can be correctly read and written by algorithms of the machine learning platform only when the role is correctly assigned.

 **Note** The machine learning platform shares service accounts with MaxCompute, because it runs on the MaxCompute framework. During authorization, you must assign the default role to your MaxCompute service account.

You can use RAM authorization to grant OSS access permissions to Machine Learning Platform for AI. Click **Settings** to grant permission to read and write OSS data. For more information, see [RAM authorization](#).

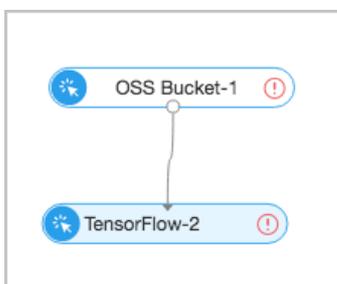
RAM authorization

1. Log on to the Machine Learning Platform For AI console, click **Settings** in the left-side navigation pane, and select **General**.
2. Under **OSS Authorization**, select **Authorize Machine Learning Platform for AI to access my OSS resources**.
3. The following page is displayed. Click **Click here to authorize access in RAM**. The RAM page is displayed.
4. Click **I Agree**.

Note To view details about the AliyunODPSPAIDefaultRole policy, log on to the [RAM console](#). The default role AliyunODPSPAIDefaultRole contains the following permission information.

Permission (Action)	Description
oss:PutObject	Upload a file or folder object.
oss:GetObject	Obtain a file or folder object.
oss:ListObjects	Query file information.
oss:DeleteObjects	Delete an object.

- Go back to the machine learning page and click **Refresh**. RAM information is automatically recorded to the components.
- Use the deep learning framework. Connect the **Read OSS Bucket** component to the corresponding deep learning component to obtain permissions to read and write OSS data.



4.4.7.3. TensorFlow 1.4

TensorFlow (TF) is an open-source machine learning framework. It is easy to use for algorithm developers. The TF framework is integrated into Apsara Stack Machine Learning Platform for AI. You can write code and adjust computing resources flexibly by using the TF compute engine. The TF computing engine is a Graphics Processing Unit (GPU) cluster.

Parameters

- Parameter settings

Parameter	Description
Python Code Files	The program execution files. Multiple files can be packaged and uploaded in the tar.gz format.
Primary Python File	Optional. The primary file in a compressed code file package.
Data Source Directory	The path of data sources. You can select Object Storage Service (OSS) data sources.
Configuration File Hyperparameters and Custom Parameters	Machine Learning Platform for AI Tensorflow allows you to use commands to pass in hyperparameter settings and try different learning rates and batch sizes during model testing.
Output Directory	The path of the output model.

- Tuning

You can specify the number of GPUs based on the complexity of jobs.

PAI command

 **Note** You do not need to set all parameters. For the definitions of these parameters, see [Parameters](#). We recommend that you do not directly copy the following command.

```
PAI -name tensorflow_ext140
-Dbuckets="oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smoke_tensorflow/mnist/"
-DgpuRequired="100"
-Darn="acs:ram::166408185518****:role/aliyunodpspaidefaultrole"
-Dscript="oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smoke_tensorflow/mnist_ext.py";
```

The following table lists the descriptions of the parameters.

Parameters

Parameter	Description	Valid values	Default value
script	Required. The TF algorithm file. This file can be a single file or compressed as a tar.gz format package.	oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smoke_tensorflow/mnist_ext.py	N/A
entryFile	Optional. The name of the primary Python file. If the script is a compressed package in the tar.gz format, this parameter is required.	train.py	Null
buckets	Required. The input OSS buckets. You can specify multiple buckets separated with commas (.). Each bucket must end with a forward slash (/).	oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smoke_tensorflow/mnist/	Null
arn	Required. The Alibaba Cloud Resource Name (ARN) of an OSS object.	N/A	Null
gpuRequired	Required. This parameter indicates the number of GPUs to be used.	200	100
checkpointDir	Optional. The TF checkpoint directory.	oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smoke_tensorflow/mnist/	Null
cluster	Optional.	A JSON format value. Quotation marks must be escaped.	Null

Parameter	Description	Valid values	Default value
hyperParameters	Optional. The path of the command line hyperparameters.	oss://imagenet.oss-cn-shanghai-internal.aliyuncs.com/smlake_tensorflow/mnist/hyper_parameters.txt	Null

- script and entryFile are used to specify the TF algorithm script to be executed. If the algorithm is complex and divided into multiple files, you can package the files into a tar.gz file and use entryFile to specify the primary Python file.
- checkpointDir is used to specify the OSS path to be written by algorithms. You must specify the OSS path when you save TensorFlow models.
- buckets is used to specify the OSS path to be read by algorithms. To use OSS, you must specify an OSS bucket.
- Distributed Machine Learning Platform for AI TensorFlow supports cluster. You can use cluster to specify the number of parameter servers and workers. cluster is in JSON format, and the quotation marks must be escaped. The JSON code must contain two keys: ps and worker. Both the ps and worker parameters contain count, gpu, cpu, and memory.

Keyword	Description	Default value	Remarks
count	Required. The number of parameter servers or workers.	-	None
gpu	Optional. The number of GPUs allocated to each parameter server or worker. 100 represents a single GPU card.	For parameter servers, the default value is 0. For workers, the default value is 100.	If the number of GPUs allocated to each worker is set to 0, Machine Learning Platform for AI will reset the value to 100 to ensure the task is scheduled properly.
cpu	Optional. The number of CPUs allocated to each parameter server or worker. 100 represents a single CPU card.	600	None
memory	The memory size allocated to each parameter server or worker. 100 represents 100 MB.	30000	None

Examples

The MNIST digit classification set is a set of handwritten digits 1 through 9 that contains training and test sets for machine learning models.

1. Upload the Python execution files and training datasets to OSS. In this case, create a bucket on OSS in China (Shanghai) and name the bucket as tfmnist001. Upload the Python script and training data.
2. Drag and drop the **Read OSS Bucket** and **TensorFlow** components onto the canvas to create the following experiment. Set the region for the OSS bucket and configure RAM authorization.
3. Set the TensorFlow parameters. Set the paths for **Python Code Files**, **Primary Python File**, and **Data**

Source Directory.

4. Click **Run** and wait for the experiment to complete running.
5. Right-click the **TensorFlow** component and view the running log.

4.4.8. Time series

4.4.8.1. x13_arima

Autoregressive Integrated Moving Average Model (ARIMA) is a well-known time series prediction method defined by Box and Jenkins in the early 1970s. This model is also called the Box-Jenkins model or the Box-Jenkins method. x13-arima is an ARIMA algorithm for seasonal adjustment based on the open-source X-13ARIMA-SEATS algorithm.

PAI command

```

pai -name x13_arima
  -project algo_public
  -DinputTableName=pai_ft_x13_arima_input
  -DseqColName=id
  -DvalueColName=number
  -Dorder=3,1,1
  -Dstart=1949.1
  -Dfrequency=12
  -Dseasonal=0,1,1
  -Dperiod=12
  -DpredictStep=12
  -DoutputPredictTableName=pai_ft_x13_arima_out_predict
  -DoutputDetailTableName=pai_ft_x13_arima_out_detail
  
```

Algorithm parameters

Parameters

Parameter	Description	Valid value	Default value
inputTableName	Required. The name of the input table.	Table name	N/A
inputTablePartitions	Optional. The partitions selected from the input table for training. in the <code>partition_name=value</code> format. To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.

Parameter	Description	Valid value	Default value
seqColName	Required. The name of the time series column.	Column name	This parameter is only used to sort the column specified by <code>valueColName</code> . The value does not affect the calculated results.
valueColName	Required. The name of the value column.	Column name	N/A
groupColNames	Optional. The name of the stratification column. Separate multiple columns with commas (,), such as <code>col0,col1</code> ; . A time series is created for each stratum.	Column name	N/A
order	Required. p , d , and q indicate the autoregressive coefficient, difference, and moving regression coefficient, respectively.	p , d , and q must be non-negative integers in the range of [0, 36].	N/A
start	Optional. The start date of a time series.	A string in the <code>year.seasonal</code> format, such as 1986.1 For more information, see the time series format section.	1.1
frequency	Optional. The frequency of a time series. Unit: months/year	A positive integer in the range of (0, 12) For more information, see the time series format section.	12
seasonal	Optional. sp , sd , and sq indicate the seasonal autoregressive coefficient, seasonal difference, and seasonal moving regression coefficient, respectively.	sp , sd , and sq must be non-negative integers in the range of [0, 36].	<code>seasonal</code> is not set by default.
period	Optional. The seasonal period.	A number in the range of (0, 100]	frequency
maxiter	Optional. The maximum number of iterations.	A positive integer	1500

Parameter	Description	Valid value	Default value
tol	Optional. The degree of tolerance.	A double type value	1e-5
predictStep	Optional. The number of prediction items.	A number in the range of (0, 365]	12
confidenceLevel	Optional. The prediction confidence level.	A number in the range of (0, 1)	0.95
outputPredictTableName	Required. The name of the output prediction table.	Table name	N/A
outputDetailTableName	Required. The name of the output detail table.	Table name	N/A
outputTablePartition	Optional. The partition in the output table.	Partition name	The output table is non-partitioned by default.
coreNum	Optional. The number of cores.	A positive integer used with memSizePerCore	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Time series format

- The start and frequency parameters specify the two time dimensions of data (valueColName): TS1 and TS2.
- The frequency parameter indicates the data frequency within a period, which equals the frequency of TS2 in each TS1.
- The start parameter must be in the `n1.n2` format. This indicates that the start date is the N2 TS2 in the N1 TS1.

Unit time	TS1	TS2	Frequency	Start date
12 months/year	Year	Month	12	1949.2 indicates the second month of year 1949.
Four quarters/year	Year	Quarter	4	1949.2 indicates the second quarter of year 1949.
Seven days/week	Day	Week	7	1949.2 indicates the second day of the 1949th week.
1	Any time unit	1	1	1949.1 indicates the 1949th (year, day, or hour).

Example: value=[1,2,3,5,6,7,8,9,10,11,12,13,14,15]

- **start=1949.3** and **frequency=12** indicate that the data frequency is monthly, and the prediction start date is 1950.06.

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949			1	2	3	4	5	6	7	8	9	10
1950	11	12	13	14	15							

- **start=1949.3** and **frequency=4** indicate that the data frequency is quarterly, and the prediction start date is 1953.02.

Year	Qtr1	Qtr2	Qtr3	Qtr4
1949			1	2
1950	3	4	5	6
1951	7	8	9	10
1952	11	12	13	14
1953	14			

- **start=1949.3** and **frequency=7** indicate that the data frequency is daily, and the prediction start date is 1951.04.

Week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
1949			1	2	3	4	5
1950	6	7	8	9	10	11	12
1951	13	14	15				

- **start=1949.1** and **frequency=1** indicate that the prediction start date is 1963.00 regardless of the time unit used.

Cycle	p1
1949	1
1950	2
1951	3
1951	4
1952	5
1953	6
1954	7
1955	8

Cycle	p1
1956	9
1957	10
1958	11
1959	12
1960	13
1961	14
1962	15

Examples

- Data used for testing: AirPassengers. The data set contains the number of passengers for international airlines each month from 1949 to 1960. It can be downloaded from <https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/AirPassengers.html>.

```
create table pai_ft_x13_arma_input(id bigint,number bigint);
tunnel upload data/airpassengers.csv pai_ft_x13_arma_input -h true;
```

- PAI command

```
pai -name x13_arma
  -project algo_public
  -DinputTableName=pai_ft_x13_arma_input
  -DseqColName=id
  -DvalueColName=number
  -Dorder=3,1,1
  -Dseasonal=0,1,1
  -Dstart=1949.1
  -Dfrequency=12
  -Dperiod=12
  -DpredictStep=12
  -DoutputPredictTableName=pai_ft_x13_arma_out_predict
  -DoutputDetailTableName=pai_ft_x13_arma_out_detail
```

- Output description

- The columns of the output table specified by outputPredictTableName are as follows.

Column	Description
pdate	The prediction date.
forecast	The prediction result.
lower	The lower threshold of the prediction result when the confidence level is specified (default value: 0.95).
upper	The upper threshold of the prediction result when the confidence level is specified (default value: 0.95).

Output data

1	196101	444.445401291661	422.354385657496	466.536416925825
2	196102	420.971087699795	394.745551231805	447.196624167785
3	196103	453.432019696644	423.435661316528	483.428378076759
4	196104	490.922534668881	458.870488854835	522.974580482926
5	196105	503.877174753018	470.308964411549	537.445385094488
6	196106	566.536076521906	531.945171132091	601.126981911721
7	196107	652.606993945368	617.2596149799	687.954372910837
8	196108	639.841497141155	603.933582941945	675.749411340364
9	196109	542.341866147189	506.000630530659	578.683101763719
10	196110	494.745102803541	458.0614903363	531.428715270782
11	196111	426.635134341211	389.672783323704	463.597485358717
12	196112	468.722280768837	431.527372120416	505.917189417259

- o The columns of the output table specified by outputDetailTableName are as follows.

Column	Description
key	"model" indicates the model. "evaluation" indicates the evaluation result. "parameters" indicates the training parameters. "log" indicates the training log.
summary	The storage details.

Output data

1	model	{ "comment": { "ma": "arima estimate", "mr": "regress...
2	evaluation	{ "comment": { "aic": "AIC", "aicc": "AICC (F-correcte...
3	paramters	{ "arima": { "d": 1, "isSeasonal": true, "p": 3, "period":...
4	log	1 Log for X-13ARIMA-SEATS program (Version 1.1...

- Model data (key=model)

operator	factor	period	lag	estimate	standard error
AR	Nonseasonal	1	1	0.6135	0.0928
AR	Nonseasonal	1	2	0.2403	0.1035
AR	Nonseasonal	1	3	-0.0732	0.0906
MA	Nonseasonal	1	1	0.9737	0.0376
MA	Seasonal	12	12	0.1051	0.1031

- Evaluation metrics (key=evaluation)

Name	Indicator
AIC	1019.6973
BIC	1036.9485
Hannan-Quinn	1026.7072
Log likelihood	-503.8487
Effective number of observations	131
Number of observations	144
variance	127.0384

4.4.8.2. x13_auto_arima

ARIMA is described in [x13_arima](#). The x13_auto_arima algorithm includes a process of automatic model selection.

The x13_auto_arima selection process is as follows:

- **Default model estimation**

In the case of frequency = 1, the default model is (0,1,1).

In the case of frequency > 1, the default model is (0,1,1)(0,1,1).

- **Identification of differencing orders**

Skip this step if you have configured diff and SeasonalDiff.

Use `Unit root test (wiki)` to determine the difference d and the seasonal difference D .

- **Identification of ARMA model orders**

Select the optimal model based on `BIC (wiki)`. The `maxOrder` and `maxSeasonalOrder` parameters are used in this step.

- **Comparison of identified model with default model**

Use `Ljung-Box Q statistic(wiki)` to compare the models. If both models are unacceptable, use the `(3,d,1)(0,D,1)` model.

- **Final model checks**

PAI command

```
pai -name x13_auto_arima
  -project algo_public
  -DinputTableName=pai_ft_x13_arima_input
  -DseqColName=id
  -DvalueColName=number
  -Dstart=1949.1
  -Dfrequency=12
  -DpredictStep=12
  -DoutputPredictTableName=pai_ft_x13_arima_out_predict2
  -DoutputDetailTableName=pai_ft_x13_arima_out_detail2
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	Table name	-
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
<code>seqColName</code>	Required. The name of the time series column.	Column name	This parameter is only used to sort <code>valueColNames</code> . It is not relevant to the calculated output.
<code>valueColName</code>	Required. The name of the value column.	Column name	-

Parameter	Description	Valid values	Default value
groupColNames	Optional. The name of the stratification column. Separate multiple columns with commas (,), such as <code>col0,col1</code> ; . A time series is created for each group.	Column name	-
start	Optional. The start date of a time series.	A string in the format of <code>year.seasonal</code> , such as <code>1986.1</code> For more information, see the time series format section.	1.1
frequency	Optional. The frequency of a time series.	A positive integer in the range of (0, 12) For more information, see the time series format section.	The frequency is 12 months/year by default.
maxOrder	Optional. The maximum values of p and q.	A positive integer in the range of [0, 4]	2
maxSeasonalOrder	Optional. The seasonal maximum values of p and q.	A positive integer in the range of [0, 2]	1
maxDiff	Optional. The maximum value of differential d.	A positive integer in the range of [0, 2]	2
maxSeasonalDiff	Optional. The maximum value of seasonal differential d.	A positive integer in the range of [0, 1]	1
diff	Optional. The differential d.	A positive integer in the range of [0, 2] If both <code>diff</code> and <code>maxDiff</code> are set, <code>maxDiff</code> is ignored. If <code>diff</code> is set, then <code>seasonalDiff</code> must also be set.	Default value: -1. This value indicates that <code>diff</code> is not specified by default.
seasonalDiff	Optional. The seasonal differential d.	A positive integer in the range of [0, 1] If both <code>seasonalDiff</code> and <code>maxSeasonalDiff</code> are set, <code>maxSeasonalDiff</code> is ignored.	Default value: -1. This value indicates that <code>seasonalDiff</code> is not specified by default.

Parameter	Description	Valid values	Default value
maxiter	Optional. The maximum number of iterations.	A positive integer	1500
tol	Optional. The degree of tolerance.	A double type value	1e-5
predictStep	Optional. The number of prediction items.	A number in the range of (0, 365]	12
confidenceLevel	Optional. The prediction confidence level.	A number in the range of (0, 1)	0.95
outputPredictTableName	Required. The name of the output prediction table.	Table name	-
outputDetailTableName	Required. The name of the output detail table.	Table name	-
outputTablePartition	Optional. The partitions in the output table.	Partition name	No partition is specified by default.
coreNum	Optional. The number of cores.	A positive integer used with memSizePerCore	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Time format

- The start and frequency parameters specify the two time dimensions of data (valueColName): TS1 and TS2.
- The frequency parameter indicates the data frequency within a period, which equals the frequency of TS2 in each TS1.
- The start parameter is in the format of `n1.n2`. This indicates that the start date is the N2 TS2 in the N1 TS1.

Unit time	ts1	ts2	Frequency	Start date
12	Year	Month	12	1949.2 indicates the second month of year 1949.
4	Year	Quarter	4	1949.2 indicates the second quarter of year 1949.
7	Day	Week	7	1949.2 indicates the second day of a week in year 1949.

Unit time	ts1	ts2	Frequency	Start date
1	Any time unit	1	1	1949.1 indicates the 1949th (year, day, or hour).

For example, value=[1,2,3,5,6,7,8,9,10,11,12,13,14,15].

- **start=1949.3** and **frequency=12** indicate that the data frequency is monthly, and the prediction start date is 1950.06.

Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949			1	2	3	4	5	6	7	8	9	10
1950	11	12	13	14	15							

- **start=1949.3** and **frequency=4** indicate that the data frequency is quarterly, and the prediction start date is 1953.02.

Year	Qtr1	Qtr2	Qtr3	Qtr4
1949			1	2
1950	3	4	5	6
1951	7	8	9	10
1952	11	12	13	14
1953	14			

- **start=1949.3** and **frequency=7** indicate that the data frequency is daily, and the prediction start date is 1951.04.

Week	Sun	Mon	Tue	Wed	Thu	Fri	Sat
1949			1	2	3	4	5
1950	6	7	8	9	10	11	12
1951	13	14	15				

- **start=1949.1** and **frequency=1** indicate that the end date is 1963.00.

Period	p1
1949	1
1950	2
1951	3
1951	4
1952	5

Period	p1
1953	6
1954	7
1955	8
1956	9
1957	10
1958	11
1959	12
1960	13
1961	14
1962	15

Examples

- Data used for testing: AirPassengers. This data set contains the number of passengers for international airlines each month from 1949 to 1960. It can be downloaded from <https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/AirPassengers.html>.

```
create table pai_ft_x13_arima_input(id bigint,number bigint);
tunnel upload data/airpassengers.csv pai_ft_x13_arima_input -h true;
```

- PAI command

```
pai -name x13_auto_arima
  -project algo_public
  -DinputTableName=pai_ft_x13_arima_input
  -DseqColName=id
  -DvalueColName=number
  -Dstart=1949.1
  -Dfrequency=12
  -DmaxOrder=4
  -DmaxSeasonalOrder=2
  -DmaxDiff=2
  -DmaxSeasonalDiff=1
  -DpredictStep=12
  -DoutputPredictTableName=pai_ft_x13_arima_auto_out_predict
  -DoutputDetailTableName=pai_ft_x13_arima_auto_out_detail
```

- Output description:

- Output table: outputPredictTableName. The columns are as follows.

Column name	Description
pdate	The prediction date.
forecast	The prediction result.
lower	The lower threshold of the prediction result when the confidence level is confidenceLevel (default value: 0.95).
upper	The upper threshold of the prediction result when the confidence level is confidenceLevel (default value: 0.95).

Data:

	key	summary
1	model	{ "comment": { "ma": "arima estimate", "mr": "regress...
2	evaluation	{ "comment": { "aic": "AIC", "aicc": "AICC (F-correcte...
3	paramters	{ "arima": { "d": 1, "isSeasonal": true, "p": 3, "period":...
4	log	1 Log for X-13ARIMA-SEATS program (Version 1.1...

- Output table: outputDetailTableName. The columns are as follows.

Column name	Description
key	"model" indicates the model. "evaluation" indicates the evaluation result. "parameters" indicates the training parameters. "log" indicates the training log.
summary	Storage details.

4.4.9. Text analysis

4.4.9.1. Word splitting

Based on Alibaba Word Segmenter (AliWS), this component performs word splitting on documents specified by columns. Segmented words are separated with spaces. If you have set the part-of-speech (POS) tagging or semantic tagging parameters, the component outputs the word splitting results, POS tagging results, and semantic tagging results. Forward slashes (/) are used as delimiters for POS tagging. Vertical bars (|) are used as delimiters for semantic tagging. Only Chinese Taobao word segmentation and Internet word segmentation are supported.

Parameter settings

Word segmentation algorithms: CRF and UNIGRAM.

Parameters

Parameter	Description
Recognition Options	Specifies whether to recognize nouns with special meanings during word splitting.
Merge Options	Considers the terms used in certain industries as a whole without splitting.
Tokenizer	Allows you to select the Taobao word segmentation or Internet word segmentation. Taobao word segmentation is recommended.
Pos Tagger	Specifies whether to mark the part of speech for each word. If this parameter is specified, the part of speech for each word is marked in the output.

Examples

The following input table consists of the id column (document IDs) and the text column (document content).

PAI command

```

pai -name split_word
-project algo_public
-DinputTableName=doc_test
-DselectedColNames=content1,content2
-DoutputTableName=doc_test_split_word
-DinputTablePartitions="region=cctv_news"
-DoutputTablePartition="region=news"
-Dtokenizer=TAOBAO_CHN
-DenableDfa=true
-DenablePersonNameTagger=false
-DenableOrganizationTagger=false
-DenablePosTagger=false
-DenableTelephoneRetrievalUnit=true
-DenableTimeRetrievalUnit=true
-DenableDateRetrievalUnit=true
-DenableNumberLetterRetrievalUnit=true
-DenableChnNumMerge=false
-DenableNumMerge=true
-DenableChnTimeMerge=false
-DenableChnDateMerge=false
-DenableSemanticTagger=true

```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	The name of the input table.	-	-
selectedColNames	The names of the columns selected from the input table for word segmentation.	Separate multiple columns with commas (,).	-

Parameter	Description	Valid values	Default value
outputTableName	The name of the output table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (.).	-	All partitions in the input table are selected by default.
outputTablePartition	The partition in the output table.	-	The output table is non-partitioned by default.
tokenizer	The type of the classifier.	TAOBAO_CHN and INTERNET_CHN	Default value: TAOBAO_CHN. <i>TAOBAO_CHN</i> represents Taobao word segmentation. <i>INTERNET_CHN</i> represents Internet word segmentation.
enableDfa	Specifies whether to enable simple entity recognition.	true and false	true
enablePersonNameTagger	Specifies whether to enable personal name recognition.	true and false	false
enableOrgnizationTagger	Specifies whether to enable organization name recognition.	true and false	false
enablePosTagger	Specifies whether to enable part-of-speech tagging.	true and false	false
enableTelephoneRetrievalUnit	Specifies whether to enable retrieval unit configuration for telephone number recognition.	true and false	true
enableTimeRetrievalUnit	Specifies whether to enable retrieval unit configuration for time ID recognition.	true and false	true

Parameter	Description	Valid values	Default value
<code>enableDateRetrievalUnit</code>	Specifies whether to enable retrieval unit configuration for date ID recognition.	true and false	true
<code>enableNumberLetterRetrievalUnit</code>	Specifies whether to enable retrieval unit configuration for number and letter recognition.	true and false	true
<code>enableChnNumMerge</code>	Specifies whether to merge Chinese numbers into a retrieval unit.	true and false	false
<code>enableNumMerge</code>	Specifies whether to merge regular numbers into a retrieval unit.	true and false	true
<code>enableChnTimeMerge</code>	Specifies whether to merge Chinese time into a semantic unit.	true and false	false
<code>enableChnDateMerge</code>	Specifies whether to merge Chinese dates into a semantic unit.	true and false	false
<code>enableSemanticTagger</code>	Specifies whether to enable semantic tagging.	true and false	false

4.4.9.2. Deprecated word filtering

Deprecated word filtering is a preprocessing method in text analysis. This method is used to filter out the noise in word splitting results, such as of, yes, and ah.

Parameter settings

The left and right input ports are as follows:

- Input table, which is a word splitting result table for filtering. Parameter: `inputTableName`
- Deprecated word table, which is a one-column table with each row containing a deprecated word. Parameter: `noiseTableName`

PAI command

```
PAI -name FilterNoise
-project algo_public
-DinputTableName="test_input"
-DnoiseTableName="noise_input"
-DoutputTableName="test_output"
-DselectedColNames="words_seg1,words_seg2"
-Dlifecycle=30
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	-	All partitions in the input table are selected by default.
noiseTableName	Required. The name of the deprecated word table.	A one-column table with each row containing a deprecated word	-
noiseTablePartitions	Optional. The partitions selected from the deprecated word table.	-	All partitions in the table are selected by default.
outputTableName	Required. The name of the output table.	-	-
selectedColNames	Required. The name of the column to be filtered. Separate multiple columns with commas (,).	-	-
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of (0, 65536)	Automatically calculated.

4.4.9.3. String similarity

String similarity calculation is a basic operation in machine learning that is used in information retrieval, natural language processing, and bioinformatics. This algorithm supports five methods to calculate similarity: Levenshtein distance, longest common substring, string subsequence kernel, cosine, and simhash_hamming. It also supports two input methods: string-to-string calculation and top N calculation.

PAI command

```
PAI -name string_similarity
-project algo_public
-DinputTableName="pai_test_string_similarity"
-DoutputTableName="pai_test_string_similarity_output"
-DinputSelectedColName1="col0"
-DinputSelectedColName2="col1";
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
outputTableName	Required. The name of the output table.	-	-
inputSelectedColName 1	Optional. The name of the first column for similarity calculation.	-	By default, the first string type column in the table is selected.
inputSelectedColName 2	Optional. The name of the second column for similarity calculation.	-	The second string type column in the table is selected by default.
inputAppendColNames	Optional. The names of columns appended to the output table.	-	No column is appended by default.
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	-	The whole table is selected by default.
outputColName	Optional. The name of the similarity column in the output table. The column name can be up to 128 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.	-	output
method	Optional. The similarity calculation method.	levenshtein, levenshtein_sim, lcs, lcs_sim, ssk, cosine, simhash_hamming, simhash_hamming_sim, minhash_sim, and hash_jaccard_sim	levenshtein_sim
lambda	Optional. The weight of the matching string. This parameter takes effect when similarityType is set to ssk.	(0, 1)	0.5
k	Optional. The length of the substring. This parameter takes effect when similarityType is set to ssk or cosine.	(0, 100)	2
kVec	Optional. The number of MinHash instances.	A positive integer	2

Parameter	Description	Valid values	Default value
b	Optional. The number of buckets.	A positive integer	1
seed	Optional. The random seed used in a MinHash instance.	A positive integer	0
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of (0, 65536)	Automatically calculated.

Examples

- SQL statement to generate data:

```
create table pai_ft_string_similarity_input
as select * from
(select 0 as id, "Beijing" as col0,
"Beijing" as col1 from dual union all
select 1 as id,
"Beijing" as col0,
"Beijing Shanghai" as col1 from dual union all
select 2 as id,
"Beijing" as col0,
"Beijing Shanghai Hongkong" as col1 from dual )tmp;
```

- PAI command

```
PAI -name string_similarity
-project sre_mpi_algo_dev
-DinputTableName=pai_ft_string_similarity_input
-DoutputTableName=pai_ft_string_similarity_output
-DinputSelectedColName1=col0
-DinputSelectedColName2=col1
-Dmethod=simhash_hamming
-DinputAppendColNames=col0,col1;
```

- Output description
 - Output obtained by using the simhash_hamming method:

col0 ▲	col1 ▲	output ▲
beijing	beijing	0
beijing	beijing shanghai	6
beijing	beijing shanghai xianggang	13

- Output obtained by using the simhash_hamming_sim method:

col0 ▲	col1 ▲	output ▲
beijing	beijing	1
beijing	beijing shanghai	0.90625
beijing	beijing shanghai xianggang	0.796875

4.4.9.4. Convert row, column, and value to KV pair

This component converts rows, columns, and values into KV pairs. A row, column, and value set is defined as XXD or XXL, where X can represent any type, D represents Double, and L represents Bigint. The row, column, and value set is converted into KV format (row,[col_id:value]). The row and value types are consistent with the original input data. The col_id type is Bigint, and the column is mapped to col_id based on the index table.

PAI command

```
PAI -name triple_to_kv
-project algo_public
-DinputTableName=test_data
-DoutputTableName=test_kv_out
-DindexOutputTableName=test_index_out
-DidColName=id
-DkeyColName=word
-DvalueColName=count
-DinputTablePartitions=ds=test1
-DindexInputTableName=test_index_input
-DindexInputKeyColName=word
-DindexInputKeyIdColName=word_id
-DkvDelimiter=:
-DpairDelimiter=;
-Dlifecycle=3
```

Parameters

Parameters

Parameter	Description	Default value
inputTableName	Required. The name of the input table.	The input table cannot be empty.
idColName	Required. The name of the column to be retained after the table is converted into a KV table.	-
keyColName	Required. The name of the key column in the KV table.	-
valueColName	Required. The name of the value column in the KV table.	-
outputTableName	Required. The name of the output KV table.	-

Parameter	Description	Default value
<code>indexOutputTableName</code>	Required. The name of the index table for the output keys.	-
<code>indexInputTableName</code>	Optional. The name of the input index table.	No index table is set by default. The table cannot be empty and it does not need to contain indexes for all of the output keys.
<code>indexInputKeyColName</code>	Optional. The name of the key column in the input index table.	No key column is specified by default. This parameter is required if <code>indexInputTableName</code> is set.
<code>indexInputKeyIdColName</code>	Optional. The name of the index column in the input index table.	No index column is specified by default. This parameter is required if <code>indexInputTableName</code> is set.
<code>inputTablePartitions</code>	Optional. The partitions in the input table.	No partition is specified by default. Only one partition can be input.
<code>kvDelimiter</code>	Optional. The delimiter used to separate the key and value.	The default delimiter is a colon (:).
<code>pairDelimiter</code>	Optional. The delimiter used to separate KV pairs.	The default delimiter is a semicolon (;).
<code>lifecycle</code>	Optional. The lifecycle of the output table.	No lifecycle is set by default.
<code>coreNum</code>	Optional. The number of cores.	-1
<code>memSizePerCore</code>	Optional. The memory size of each core. Valid values: 100 to 65536.	-1

Examples

- SQL statement to generate data:

```
drop table if exists triple2kv_test_input;
create table triple2kv_test_input as
select * from
( select '01' as id, 'a' as word,
  10 as count from dual union all
  select '01' as id, 'b' as word,
  20 as count from dual union all
  select '01' as id, 'c' as word,
  30 as count from dual union all
  select '02' as id,
  'a' as word,
  100 as count from dual union all
  select '02' as id, 'd' as word,
  200 as count from dual union all
  select '02' as id, 'e' as word,
  300 as count from dual ) tmp;
```

- PAI command

```
PAI -name triple_to_kv  
-project algo_public  
-DinputTableName=triple2kv_test_input  
-DoutputTableName=triple2kv_test_input_out  
-DindexOutputTableName=triple2kv_test_input_index_out  
-DidColName=id  
-DkeyColName=word  
-DvalueColName=count  
-Dlifecycle=1;
```

Input description

Input table

Input description

id	word	count
01	a	10
01	b	20
01	c	30

Output description

- The output KV table is as follows, where custom KV delimiters can be used. Output description

id	key_value
01	1:10;2:20;3:30

- The output index table that contains indexes for the words is as follows. Output index table

key	key_id
a	1
b	2
c	3

4.4.9.5. String similarity - Top N

String similarity calculation is a basic operation in machine learning that is used in information retrieval, natural language processing, and bioinformatics. This algorithm supports five methods to calculate similarity: Levenshtein distance, longest common substring, string subsequence kernel, cosine, and simhash_hamming. It also supports two input methods: string-to-string calculation and top N calculation.

PAI command

```
PAI -name string_similarity_topn
-project algo_public
-DinputTableName="pai_test_string_similarity_topn"
-DoutputTableName="pai_test_string_similarity_topn_output"
-DmapTableName="pai_test_string_similarity_map_topn"
-DinputSelectedColName="col0"
-DmapSelectedColName="col1";
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
mapTableName	Required. The name of the mapping table.	-	-
outputTableName	Required. The name of the output table.	-	-
inputSelectedColName	Optional. The name of the column selected from the left table for similarity calculation.	-	The first string type column in the table is selected by default.
mapSelectedColName	Optional. The name of the column selected from the mapping table for similarity calculation. The similarities between each row in the left table and all strings in the mapping table are calculated, and the top N entries are output.	-	The first string type column in the table is selected by default.
inputAppendColNames	Optional. The names of columns appended to the output table from the input table.	-	No column is appended by default.
inputAppendRenameColNames	Optional. The aliases of columns appended to the output table from the input table. This parameter takes effect when <code>inputAppendColNames</code> is specified.	-	No alias is specified by default.
mapAppendColNames	Optional. The names of columns appended to the output table from the mapping table.	-	No column is appended by default.

Parameter	Description	Valid values	Default value
mapAppendRenameColNames	Optional. The aliases of columns appended to the output table from the mapping table.	-	No alias is specified by default.
inputTablePartitions	Optional. The partitions selected from the input table.	-	The whole table is selected by default.
mapTablePartitions	Optional. The partitions in the mapping table.	-	The whole table is selected by default.
outputColName	Optional. The name of the similarity column in the output table. The column name can be up to 128 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.	-	output
method	Optional. The similarity calculation method.	levenshtein_sim, lcs_sim, ssk, cosine, simhash_hamming_sim, minhash_sim, and hash_jaccard_sim	levenshtein_sim
lambda	Optional. The weight of the matching string. This parameter takes effect when similarityType is set to ssk.	(0, 1)	0.5
k	Optional. The length of the substring. This parameter takes effect when similarityType is set to ssk or cosine.	(0, 100)	2
kVec	Optional. The number of MinHash instances.	A positive integer	2
b	Optional. The number of buckets.	A positive integer	1
seed	Optional. The random seed used in a MinHash instance.	A positive integer	0
topN	Optional. The number of similarity maximums to be output.	(0, +∞)	10
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Parameter	Description	Valid values	Default value
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of (0, 65536)	Automatically calculated.

Examples

- SQL statement to generate data:

```
create table pai_ft_string_similarity_topn_input
as select * from
(select 0 as id,
"Beijing" as col0 from dual union all
select 1 as id,
"Beijing Shanghai" as col0 from dual union all
select 2 as id,
"Beijing Shanghai Hongkong" as col0 from dual )tmp;
```

- PAI command

```
PAI -name string_similarity_topn
-project sre_mpi_algo_dev
-DinputTableName=pai_ft_string_similarity_topn_input
-DmapTableName=pai_ft_string_similarity_topn_input
-DoutputTableName=pai_ft_string_similarity_topn_output
-DinputSelectedColName=col0
-DmapSelectedColName=col0
-DinputAppendColNames=col0
-DinputAppendRenameColNames=input_col0
-DmapAppendColNames=col0
-DmapAppendRenameColNames=map_col0
-Dmethod=simhash_hamming_sim;
```

- Output.

input_col0 ▲	map_col0 ▲	output ▲
beijing	beijing	1
beiji beijing	beijing shanghai	0.90625
beijing	beijing shanghai xianggang	0.796875
beijing shanghai	beijing shanghai	1
beijing shanghai	beijing	0.90625
beijing shanghai	beijing shanghai xianggang	0.828125
beijing shanghai xianggang	beijing shanghai xianggang	1
beijing shanghai xianggang	beijing shanghai	0.828125
beijing shanghai xianggang	beijing	0.796875

4.4.9.6. N-gram counting

N-gram counting is a step in language model training. N-grams are generated based on words. The number of the corresponding N-grams in all corpora is counted. The N-gram counting model counts the number of N-grams in all documents rather than in a single document. For more information, see [ngram-count](#).

PAI command

```
PAI -name ngram_count
  -project algo_public
  -DinputTableName=pai_ngram_input
  -DoutputTableName=pai_ngram_output
  -DinputSelectedColNames=col0
  -DweightColName=weight
  -DcoreNum=2
  -DmemSizePerCore=1000;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	N/A
outputTableName	Required. The name of the output table.	Table name	N/A
inputSelectedColNames	Optional. The names of columns selected from the input table.	Column name	The first character type column is selected by default.
weightColName	Optional. The name of the weight column.	Column name	1
inputTablePartitions	Optional. The partitions selected from the input table.	Partition name	The whole table is selected by default.
countTableName	Optional. The name of the former N-gram counting output table. This table is merged into the output result.	Table name	N/A
countWordColName	Optional. The name of the word column in the counting table.	Column name	The second column is selected by default.
countCountColName	Optional. The name of the counting column in the counting table.	Column name	The third column is selected by default.
countTablePartitions	Optional. The partitions in the counting table.	Partition name	N/A

Parameter	Description	Valid values	Default value
<code>vocabTableName</code>	Optional. The name of the bag-of-words table. The words that are not contained in the bag-of-words table are marked with \<unk\.	Table name	N/A
<code>vocabSelectedColumnName</code>	Optional. The name of the bag-of-words column.	Column name	The first character type column is selected by default.
<code>vocabTablePartitions</code>	Optional. The partitions in the bag-of-words table.	Partition name	N/A
<code>order</code>	Optional. The maximum length of N-grams.	N/A	3
<code>lifecycle</code>	Optional. The lifecycle of the output table.	A positive integer	N/A
<code>coreNum</code>	Optional. The number of cores.	A positive integer	N/A
<code>memSizePerCore</code>	Optional. The memory size of each core.	A positive integer	N/A

4.4.9.7. Text summarization

Automatic summarization uses computers to automatically extract summaries from a source document. A summary is a simple, concise, and short document that completely and accurately describes the content of a certain document. This TextRank-based algorithm generates summaries by extracting existing sentences in the document.

PAI command

```
PAI -name TextSummarization
-project algo_public
-DinputTableName="test_input"
-DoutputTableName="test_output"
-DdocIdCol="doc_id"
-DsentenceCol="sentence"
-DtopN=2
-Dlifecycle=30;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	-	-

Parameter	Description	Valid values	Default value
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	-	All partitions in the input table are selected by default.
outputTableName	Required. The name of the output table.	-	-
docIdCol	Required. The name of the document ID column.	-	-
sentenceCol	Required. The sentence column.	Only one column can be specified.	-
topN	Optional. The top N key sentences to be output.	-	3
similarityType	Optional. The method used to calculate sentence similarity.	lcs_sim, levenshtein_sim, cosine, and ssk	lcs_sim
lambda	Optional. The weight of the matching string. This parameter takes effect when similarityType is set to ssk.	(0, 1)	0.5
k	Optional. The length of the substring. This parameter takes effect when similarityType is set to ssk or cosine.	(0, 100)	2
dampingFactor	Optional. The damping factor.	(0, 1)	0.85
maxIter	Optional. The maximum number of iterations.	[1, +]	100
epsilon	Optional. The convergence coefficient.	(0, ∞)	0.000001
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	A positive integer	Automatically calculated.

The sentence similarity options are as follows:

- **lcs_sim**: The formula is $1.0 - (\text{Length of the longest common subsequence}) / \max(\text{len}(A), \text{len}(B))$.
- **levenshtein_sim**: The formula is $1.0 - (\text{Levenshtein distance}) / \max(\text{len}(A), \text{len}(B))$.
- **cosine**: See Lodhi, Huma; Saunders, Craig; Shawe-Taylor, John; Cristianini, Nello;

Watkins. Chris (2002). "Text classification using string kernels". *Journal of Machine Learning Research*: 419-444 .

- ssk: See Leslie. C.; Eskin. E.; Noble. W.S. (2002). *The spectrum kernel: A string kernel for SVM protein classification* 7, pp. 566-575 .

 Note A and B indicate two strings, and len(A) indicates the length of string A.

Output format description

The output table contains the doc_id and abstract columns, as shown in [Output table example](#).

Output table example

doc_id	abstract
1000894	In 2008, the Shanghai Stock Exchange published disclosure guidelines for the corporate social responsibility of listed companies. Three types of companies were urged to disclose their CSR reports, and other qualified listed companies were encouraged to voluntarily disclose their CSR reports. In 2012, a total of 379 listed companies making up a 40% of all listed companies disclosed CSR reports. Of those companies, 305 were mandated to disclose CSR reports and and 75 voluntarily disclosed CSR reports. According to Hu Ruyin, Shanghai Stock Exchange will explore how to expand the scope of CSR report disclosure, revise and refine the guidelines on disclosure of the CSR reports, and encourage more organizations to promote CSR product innovation.

4.4.9.8. Keyword extraction

Keyword extraction is one of the important technologies in natural language processing. It is used to extract keywords from a document. This algorithm is based on TextRank, a variation of the PageRank algorithm used to describe the relationship between webpages. This algorithm uses the relationship between certain words to construct a network, calculate the importance of each word, and determine words with larger weights as keywords.

PAI command

```
PAI -name KeywordsExtraction
-DinputTableName=maple_test_keywords_basic_input
-DdocIdCol=docid -DdocContent=word
-DoututTableName=maple_test_keywords_basic_output
-DtopN=19;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
-----------	-------------	--------------	---------------

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions are selected by default.
outputTableName	Required. The name of the output table.	-	-
docIdCol	Required. The name of the document ID column.	Only one column can be specified.	-
docContent	Required. The word column.	Only one column can be specified.	-
topN	Optional. The number of top N keywords to be output. If this number is smaller than the number of keywords, all keywords are output.	-	5
windowSize	Optional. The window size of the TextRank algorithm.	-	2
dumpingFactor	Optional. The damping factor of the TextRank algorithm.	-	0.85
maxIter	Optional. The maximum number of iterations of the TextRank algorithm.	-	100
epsilon	Optional. The convergence residual threshold of the TextRank algorithm.	-	0.000001
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.

Parameter	Description	Valid values	Default value
coreNum	Optional. The number of cores.	This parameter is used with memSizePerCore. The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

Examples

The words in the input table are separated with spaces, and deprecated words and all punctuations are filtered out.

Examples

docid: string	word: string
---------------	--------------

docid: string	word: string
doc0	<p>The blended-wing-body aircraft is a new direction for the future development in the aviation field Many research institutions inside and outside China have carried out research on the blended-wing-body aircraft while its fully automated shape optimization algorithm has become a new hot topic Based on the existing research achievements inside and outside China common modeling and flow solver tools have been analyzed and compared The geometric modeling grid flow field solver and shape optimization modules have been designed The pros and cons between different algorithms have been compared to achieve the optimized shape of the blended-wing-body aircraft in the conceptual design stage Geometric modeling and grid generation module are achieved based on the transfinite interpolation algorithm and spline based grid generation method The flow solver module includes the finite difference solver the finite element solver and the panel method solver The finite difference solver includes mathematical modeling of the potential flow the derivation of the Cartesian grid based variable step length difference scheme Cartesian grid generation and indexing algorithm the Cartesian grid based Neumann boundary conditions expression form derivation are achieved based on finite element difference solver The aerodynamic parameters of a two-dimensional airfoil are calculated based on the finite difference solver The finite element solver includes potential flow modeling based on the variational principle of the finite element theory the derivation of the two-dimensional finite element Kutta conditional least squares based speed solving algorithm Gmsh based two-dimensional field grid generator of airfoil with wakes design The aerodynamic parameters of a two-dimensional airfoil are calculated based on the finite element solver The panel method solver includes modeling and automatic wake generation the design of the three-dimensional flow solver of the blended-wing-body drag estimation based on the Blasius solution solver implemented in the Fortran language a mixed compilation of Python and Fortran OpenMP and CUDA based acceleration algorithm The aerodynamic parameters of a three-dimensional wing body are calculated based on the panel method solver The shape optimization module includes free form deformation algorithm genetic algorithms differential evolution algorithm Aircraft surface area calculation algorithm is based on the moments integration algorithm The volume of an aircraft calculation algorithm is based on VKT data visualization format tool</p>

PAI command

```
PAI -name KeywordsExtraction
-DinputTableName=maple_test_keywords_basic_input
-DdocIdCol=docid -DdocContent=word
-DoutputTableName=maple_test_keywords_basic_output
-DtopN=19;
```

Input/output description

Output table description

docid	keywords	weight
doc0	Based on	0.041306752223538405
doc0	Algorithm	0.03089845626854151
doc0	Modeling	0.021782865850562882
doc0	Grid	0.020669749212693957
doc0	Solver	0.020245609506360847
doc0	Aircraft	0.019850761705313365
doc0	Research	0.014193732541852615
doc0	Finite element	0.013831122054200538
doc0	Solving	0.012924593244133104
doc0	Module	0.01280216562287212
doc0	Derivation	0.011907588923852495
doc0	Shape	0.011505456605632607
doc0	Difference	0.011477831662367547
doc0	Flow	0.010969269350293957
doc0	Design	0.010830986516637251
doc0	Implementation	0.010747536556701583
doc0	Two-dimensional	0.010695570768457084
doc0	Development	0.010527342662670088
doc0	New	0.010096978306668461

4.4.9.9. Sentence splitting

You can split sentences in a document by punctuation. This component is used to preprocess text summarizations. It splits text such that each row contains only a single sentence.

PAI command

```
PAI -name SplitSentences
-project algo_public
-DinputTableName="test_input"
-DoutputTableName="test_output"
-DdocIdCol="doc_id"
-DdocContent="content"
-Dlifecycle=30
```

Parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	-	All partitions in the input table are selected by default.
outputTableName	Required. The name of the output table.	-	-
docIdCol	Required. The name of the document ID column.	-	-
docContent	Required. The name of the document content column.	Only one column can be specified.	-
delimiter	Optional. A set of characters used to determine the end of a sentence.	-	The default delimiter set contains the period (.), question mark (!), and exclamation mark (?).
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	A positive integer	Automatically calculated.

Output format description

The output table contains the doc_id and sentence columns, as shown in [Output table example](#).

Output table example

doc_id	sentence
--------	----------

doc_id	sentence
1000894	In 2008, the Shanghai Stock Exchange published disclosure guidelines for the corporate social responsibility of listed companies. Three types of companies were urged to disclose their CSR reports, and other qualified listed companies were encouraged to voluntarily disclose their CSR reports.
1000894	In 2012, a total of 379 listed companies making up a 40% of all listed companies disclosed CSR reports. Of those companies, 305 were mandated to disclose CSR reports and and 75 voluntarily disclosed CSR reports.

4.4.9.10. Semantic vector distance

You can calculate the extension words or sentences of the specified words or sentences based on the calculated semantic vectors, such as word vectors calculated by the Word2Vec component. The extension words or sentences are a set of vectors closest to a certain vector. The following example shows how to generate a list of words that are most similar to the word that you entered based on the word vectors calculated by the Word2Vec component.

PAI command

```
PAI -name SemanticVectorDistance
-project algo_public
-DinputTableName="test_input"
-DoutputTableName="test_output"
-DidColName="word"
-DvectorColNames="f0,f1,f2,f3,f4,f5"
-Dlifecycle=30
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for calculation.	-	All partitions in the input table are selected by default.
outputTableName	Required. The name of the output table.	-	-
idTableName	Optional. The name of the vector ID table for vector calculation. The table contains only one column and each row stores a vector ID.	-	No vector ID table is specified by default. This means that all vectors in the input table are calculated.

Parameter	Description	Valid values	Default value
idTablePartitions	Optional. The partitions selected from the ID table for calculation.	-	All partitions are selected by default.
idColName	Required. The name of the ID column.	-	3
vectorColNames	Optional. A list of vector column names, such as f1, f2,...	-	-
topN	Optional. The number of the closest vectors to output.	[1, +∞]	5
distanceType	Optional. The distance calculation method.	euclidean, cosine, and manhattan	euclidean
distanceThreshold	Optional. The distance threshold. Only the distances between two vectors that do not exceed this threshold are output.	(0, +∞)	∞
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core.	A positive integer	Automatically calculated.

Examples

The output table contains the original_id, near_id, distance, and rank columns.

original_id	near_id	distance	rank
hello	hi	0.2	1
hello	xxx	xx	2
Man	Woman	0.3	1
Man	xx	xx	2
..

4.4.9.11. Document similarity

This algorithm calculates the similarity between two text documents by comparing the similarities of documents or sentences separated by spaces. This algorithm's functions are similar to how the similarity of strings is calculated.

PAI command

```
PAI -name doc_similarity
-project algo_public
-DinputTableName="pai_test_doc_similarity"
-DoutputTableName="pai_test_doc_similarity_output"
-DinputSelectedColName1="col0"
-DinputSelectedColName2="col1"
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	-	-
outputTableName	Required. The name of the output table.	-	-
inputSelectedColName 1	Optional. The name of the first column for similarity calculation.	-	By default, the first string type column in the table is selected.
inputSelectedColName 2	Optional. The name of the second column for similarity calculation.	-	The name of the second string type column in the table is selected by default.
inputAppendColNames	Optional. The names of columns appended to the output table.	-	No column is appended by default.
inputTablePartitions	Optional. The partitions selected from the input table.	-	The whole table is selected by default.
outputColName	Optional. The name of the similarity column in the output table. The column name can be up to 128 characters in length and can contain letters, digits, and underscores (_). It must start with a letter.	-	output
method	Optional. The similarity calculation method.	levenshtein, levenshtein_sim, lcs, lcs_sim, ssk, cosine, simhash_hamming, and simhash_hamming_sim	levenshtein_sim
lambda	Optional. The weight of the matching word pair. This parameter takes effect if similarityType is set to ssk.	(0, 1)	0.5

Parameter	Description	Valid values	Default value
k	Optional. The length of the substring. This parameter takes effect if similarityType is set to ssk or cosine.	(0, 100)	2
kVec	Optional. The number of MinHash instances.	A positive integer	2
b	Optional. The number of buckets.	A positive integer	1
seed	Optional. The random seed used in a MinHash instance.	A positive integer	0
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size of each core. Unit: MB.	A positive integer in the range of (0, 65536)	Automatically calculated.

Examples

- SQL statement to generate data:

```
drop table if exists pai_doc_similarity_input;
create table pai_doc_similarity_input as
select * from (
select 0 as id,
"Beijing and Shanghai" as col0,
"Beijing and Shanghai" as col1 from dual union all
select 1 as id,
"Beijing and Shanghai" as col0,
"Beijing, Shanghai, and Hong Kong" as col1 from dual )tmp;
```

- PAI command

```
drop table if exists pai_doc_similarity_output;
PAI -name doc_similarity
-project algo_public
-DinputTableName=pai_doc_similarity_input
-DoutputTableName=pai_doc_similarity_output
-DinputSelectedColName1=col0
-DinputSelectedColName2=col1
-Dmethod=levenshtein_sim
-DinputAppendColNames=id,col0,col1;
```

- Input description: pai_doc_similarity_input

ID	col0	col1
1	Beijing and Shanghai	Beijing, Shanghai, and Hong Kong
0	Beijing and Shanghai	Beijing and Shanghai

- Output description: pai_doc_similarity_output

ID	col0	col1	Output
1	Beijing and Shanghai	Beijing, Shanghai, and Hong Kong	0.6666666666666667
0	Beijing and Shanghai	Beijing and Shanghai	1.0

4.4.9.12. PMI

Mutual information (MI) is a measure of information in the information theory. It can be regarded as the amount of information contained in a random variable about another variable, or the reduction in uncertainty of a random variable due to the known random variable.

This algorithm is used to count the co-occurrence of all words in several documents and calculate the point mutual information (PMI) . PMI definition: $PMI(x,y)=\ln(p(x,y)/(p(x)p(y)))=\ln(\#(x,y)D/(\#x\#y))$.

- $\#(x,y)$ indicates the number of pair(x,y).
- D indicates the total number of pairs.
- If x and y appear in the same window, the output is $\#x+=1;\#y+=1;\#(x,y)+=1$.

PAI command

```
PAI -name PointwiseMutualInformation
  -project algo_public
  -DinputTableName=maple_test_pmi_basic_input
  -DdocColName=doc
  -DoutputTableName=maple_test_pmi_basic_output
  -DminCount=0
  -DwindowSize=2
  -DcoreNum=1
  -DmemSizePerCore=110;
```

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
outputTableName	Required. The name of the output table.	Table name	-

Parameter	Description	Valid values	Default value
docColName	Required. The name of the document column after word splitting, where words are separated with spaces.	Column name	-
windowSize	Optional. The window size. For example, the value 5 refers to the five words adjacent on the right of the current word. Words that appear in the window are considered related to the current word.	[1, sentence length]	The whole row is selected by default.
minCount	The minimum word truncation frequency. Words that appear for a number of times less than this value are filtered out.	[0, 2e63]	5
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	Partition name	All partitions are selected by default.
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set by default.
coreNum	Optional. The number of cores.	This parameter is used with <code>memSizePerCore</code> . The value must be a positive integer in the range of [1, 9999].	Automatically calculated.
memSizePerCore	The memory size of each core. Unit: MB.	A positive integer in the range of [1024, 65536]	Automatically calculated.

Examples

- Data generation

<code>doc:string</code>
<code>w1 w2 w3 w4 w5 w6 w7 w8 w8 w9</code>

doc:string
w1 w3 w5 w6 w9
w0
w0 w0
w9 w1 w9 w1 w9

• PAI command

```
PAI -name PointwiseMutualInformation
  -project algo_public
  -DinputTableName=maple_test_pmi_basic_input
  -DdocColName=doc
  -DoutputTableName=maple_test_pmi_basic_output
  -DminCount=0
  -DwindowSize=2
  -DcoreNum=1
  -DmemSizePerCore=110;
```

• Output description

word1	word2	word1_count	word2_count	co_occurrences_count	pmi
w0	w0	2	2	1	2.0794415416798357
w1	w1	10	10	1	-1.1394342831883648
w1	w2	10	3	1	0.06453852113757116
w1	w3	10	7	2	-0.08961215868968704
w1	w5	10	8	1	-0.916290731874155
w1	w9	10	12	4	0.06453852113757116
w2	w3	3	7	1	0.4212134650763035
w2	w4	3	4	1	0.9808292530117262
w3	w4	7	4	1	0.13353139262452257

word1	word2	word1_count	word2_count	co_occurrences_count	pmi
w3	w5	7	8	2	0.13353139262452257
w3	w6	7	7	1	-0.42608439531090014
w4	w5	4	8	1	0
w4	w6	4	7	1	0.13353139262452257
w5	w6	8	7	2	0.13353139262452257
w5	w7	8	4	1	0
w5	w9	8	12	1	-1.0986122886681098
w6	w7	7	4	1	0.13353139262452257
w6	w8	7	7	1	-0.42608439531090014
w6	w9	7	12	1	-0.9650808960435872
w7	w8	4	7	2	0.8266785731844679
w8	w8	7	7	1	-0.42608439531090014
w8	w9	7	12	2	-0.2719337154836418
w9	w9	12	12	2	-0.8109302162163288

4.4.9.13. Word frequency statistics

Based on the word splitting results, this component outputs the words in their original order and calculates the frequency that a word occurs in the document (docContent) specified by the document ID column (docId).

Parameter settings

Input parameters: docId column and docContent column generated by the Word Splitting component.

Two output parameters:

- Output port 1: The output table contains the id, word, and count columns.
count: indicates the frequency that a word occurs in each document.
- Output port 2: The output table contains the id and word columns.

The table output by the second output port lists words in order of occurrence in the document. The table does not calculate the frequency of the occurrence. Therefore, a word may have multiple table entries in the same document. The output table format is compatible with the Word2Vec component.

Examples

In the Alibaba Cloud word splitting data, the two columns in the output table are used as the input parameters for word frequency calculation.

- Select the docId column: id.
- Select the docContent column: After the word frequency calculation is performed, the result is displayed by output port 1 on this component.

PAI command

```
pai -name doc_word_stat
    -project algo_public
    -DinputTableName=doc_test_split_word
    -DdocId=id
    -DdocContent=content
    -DoutputTableNameMulti=doc_test_stat_multi
    -DoutputTableNameTriple=doc_test_stat_triple
    -DinputTablePartitions="region=cctv_news"
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	The name of the input table.	-	-
docId	The name of the document ID column.	Only one column can be specified.	-
docContent	The name of the document content column.	Only one column can be specified.	-
outputTableNameMulti	The name of the output table that lists words in the document content after word splitting. Documents are specified by the docId column and their contents are specified by the docContent column. The words are listed in the order that they occur within the documents.	-	-

Parameter	Description	Valid values	Default value
outputTableNameTriple	The name of the output table that lists the words and the frequency of the occurrence of these words in the documents. The documents are specified by the docId column and their contents are specified by the docContent column.	-	-
inputTablePartitions	Optional. The partitions selected from the input table for training, in the format of <code>partition_name=value</code> . To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	-	All partitions in the input table are selected by default.

4.4.9.14. TF-IDF

Term frequency-inverse document frequency (TF-IDF) is typically used as a weighting technology in information retrieval and text mining. TF-IDF is a statistical method to evaluate the importance of a word for a document in a collection or corpus. The importance of a word increases as the frequency that it occurs within the document increases. The importance decreases as the frequency that the word occurs in the corpus increases. TF-IDF is frequently used by search engines as a tool in scoring and ranking the correlation between documents and user queries.

- For more information, see TF-IDF in Wikipedia.
- The TF-IDF component is used to calculate the TF-IDF value of each word that appears in a collection of documents based on word frequency statistics.

Examples

The output table in the example of the word frequency statistics component is used as the input table for the TF-IDF component. The corresponding parameter settings are as follows:

- Select the document ID column: id
- Select the word column: word
- Select the word count column: count

The output table contains the following columns: docid, word, word_count (frequency that a certain word occurs in the current document), total_word_count (total number of words in the current document), doc_count (total number of documents that contain the current word), total_doc_count (total number of documents), tf, idf.

PAI command

```
pai -name tfidf
  -project algo_public
  -DinputTableName=rgdoc_split_triple_out
  -DdocIdCol=id
  -DwordCol=word
  -DcountCol=count
  -DoutputTableName=rg_tfidf_out;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
<code>inputTableName</code>	Required. The name of the input table.	Table name	-
<code>inputTablePartitions</code>	Optional. The partitions selected from the input table for word splitting.	This value must be in the <code>partition_name=value</code> format. To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	All partitions in the input table are selected by default.
<code>docIdCol</code>	Required. The name of the document ID column.	Only one column can be specified.	-
<code>wordCol</code>	Required. The name of the word column.	Only one column can be specified.	-
<code>countCol</code>	Required. The name of the count column.	Only one column can be specified.	-
<code>outputTableName</code>	Required. The name of the output table.	Table name	-
<code>outputTablePartition</code>	The partitions in the output table.	Partition name	The output table is non-partitioned by default.

4.4.9.15. PLDA

Latent Dirichlet Allocation (LDA) is a topic model that outputs the topic of a document. It outputs the topic of each document based on probability distribution. LDA is an unsupervised learning algorithm that does not require a manually tagged training set. Instead, it only requires a set of documents and the number of topics (k). It is used in text mining, including text topic recognition, text classification, and text similarity calculation.

Parameter settings

Parameters

Parameter	Description
Topics	The number of topics output by LDA.
Alpha	The AlphaPrior Dirichlet distribution parameter of $P(z/d)$.
Beta	The AlphaPrior Dirichlet distribution parameter of $P(w/z)$.
Burn-in Iterations	The number of burn-in iterations. The parameter value must be less than the total number of iterations. The default value is 100.

Parameter	Description
Total Iterations	Optional. The total number of iterations. The parameter value must be a positive integer. The default value is 150.

Note z represents topics, w represents words, and d represents documents.

Input and output settings

Input:

The data must be in the sparse matrix format. For more information about the format, see the data format description section. You can use the Convert Row, Column, and Value to KV Pair component to convert the data. Input format as shows the following picture.

```

+-----+-----+
| id      | features |
+-----+-----+
| 2       | 38:3.0,39:1.0,40:3.0,41:1.0,42:1.0,43:2.0,44:1.0,45:1.0,46:1.0,47:1.0,48:1.0,49:2.0,50:1.0,51:1.0,52:1.0,53:1.0,54:1.0,55:1.0,56:1.0,57:1.0,58:1.0,59:1.0,60:1.0,61:1.0,62:1.0,63:1.0,64:1.0,65:1.0,66:1.0,67:1.0,68:1.0,69:1.0,70:1.0,71:1.0,72:1.0,73:1.0,74:1.0,75:1.0,76:1.0,77:2.0 |
| 1       | 0:1.0,1:2.0,3:1.0,4:1.0,5:1.0,6:1.0,7:1.0,8:1.0,9:1.0,10:1.0,11:1.0,12:1.0,13:1.0,14:2.0,15:1.0,16:1.0,17:1.0,18:1.0,19:1.0,20:1.0,21:1.0,22:1.0,23:1.0,24:1.0,25:1.0,26:1.0,27:1.0,28:1.0,29:1.0,30:1.0,31:1.0,32:1.0,33:1.0,34:1.0,35:1.0,36:2.0,39:2.0,77:3.0 |
+-----+-----+
    
```

- o Column 1: the ID of a document.
- o Column 2: KV data of the word and how frequently it occurs.

Output:

The following tables are generated in sequence: topic-word frequency contribution table, P(w/z) table, P(z/w) table, P(d/z) table, P(z/d) table, and P(z) table.

The following picture shows the output format of the topic-word frequency contribution table.

```

+-----+-----+-----+
| wordid | topic_0 | topic_1 |
+-----+-----+-----+
| 0      | 1       | 0       |
| 1      | 2       | 0       |
| 2      | 0       | 0       |
| 3      | 1       | 0       |
| 4      | 1       | 0       |
| 5      | 1       | 0       |
| 6      | 1       | 0       |
| 7      | 1       | 0       |
| 8      | 0       | 1       |
| 9      | 1       | 0       |
| 10     | 1       | 0       |
| 11     | 1       | 0       |
| 12     | 1       | 0       |
+-----+-----+-----+
    
```

PAI command

```

pai -name PLDA
    -project algo_public
    -DinputTableName=lda_input
    -DtopicNum=10
    -topicWordTableName=lda_output;
    
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
-----------	-------------	--------------	---------------

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartitions	Optional. The partitions selected from the input table for word splitting.	This value must be in the partition_name=value format. To specify multiple partitions, use the following format: name1=value1/name2=value2 . Separate multiple partitions with commas (,).	All partitions in the input table are selected by default.
selectedColNames	Optional. The names of the columns selected from the input table for LDA.	Separate multiple columns with commas (,).	All columns in the input table are selected by default.
topicNum	Required. The number of topics.	[2, 500]	-
kvDelimiter	Optional. The delimiter used to separate the key and value.	Space, comma (,), and colon (:)	The default delimiter is a colon (:).
itemDelimiter	Optional. The delimiter used to separate keys.	Space, comma (,), and colon (:)	The default delimiter is a space.
alpha	Optional. The prior Dirichlet distribution parameter of $P(z/d)$.	$(0, \infty)$	0.1
beta	Optional. The prior Dirichlet distribution parameter of $P(w/z)$.	$(0, \infty)$	0.01
topicWordTableName	Required. The name of the topic-word frequency contribution table.	Table name	-
pwzTableName	Optional. The name of the $P(w/z)$ table.	Table name	No $P(w/z)$ table is output by default.
pzwTableName	Optional. The name of the $P(z/w)$ table.	Table name	No $P(z/w)$ table is output by default.
pdzTableName	Optional. The name of the $P(d/z)$ table.	Table name	No $P(d/z)$ table is output by default.
pzdTableName	Optional. The name of the $P(z/d)$ table.	Table name	No $P(z/d)$ table is output by default.
pzTableName	Optional. The name of the $P(z)$ table.	Table name	No $P(z)$ table is output by default.

Parameter	Description	Valid values	Default value
burnIterations	Optional. The number of burn-in iterations.	A positive integer	This value must be smaller than the total number of iterations. The default value is 100.
totalIterations	Optional. The number of iterations.	A positive integer	150

4.4.9.16. Word2Vec

Word2Vec is an open-source algorithm used to convert words into vectors. By training neural networks, Word2Vec can map words to K-dimensional space vectors and map word vectors to semantics.

For information about the Google Word2Vec toolkit, visit <https://code.google.com/p/word2vec/>.

Parameter settings

- Dimension of Word Features: We recommend a value from 0 to 1000.
- Downsampling Threshold: We recommend a value from 1e-3 to 1e-5.
- Input: inputs a word column and a vocabulary.
- Output: generates a word vector table and a vocabulary.

PAI command

```
pai -name Word2Vec
    -project algo_public
    -DinputTableName=w2v_input
    -DwordColName=word
    -DoutputTableName=w2v_output;
```

Algorithm parameters

Parameters

Parameter	Description	Valid values	Default value
inputTableName	Required. The name of the input table.	Table name	-
inputTablePartitions	Optional. The partitions selected from the input table for word splitting.	The parameter value must be in the <code>partition_name=value</code> format. To specify multiple partitions, use the following format: <code>name1=value1/name2=value2</code> . Separate multiple partitions with commas (,).	All partitions in the input table are selected by default.

Parameter	Description	Valid values	Default value
wordColName	Required. The name of the word column. Each row in the word column contains a single word. </s> is used to break lines in the corpus.	Column name	-
inVocabularyTableName	Optional. The name of the input word list, which contains the wordcount output of inputTableName.	Table name	Word count is performed for the input table by default.
inVocabularyPartitions	Optional. The partitions in the input word list.	Partition name	By default, all partitions in the table specified by inVocabularyTableName are selected.
layerSize	Optional. The dimension of word features.	0 to 1000	100
cbow	Optional. The language model.	1: cbow. 0: skip-gram.	0
window	Optional. The size of the word window.	A positive integer	5
minCount	Optional. The minimum frequency of word truncation.	A positive integer	5
hs	Optional. This parameter specifies whether to use hierarchical softmax.	1: Hierarchical softmax is used. 0: Hierarchical softmax is not used.	1
negative	Optional. The negative sampling.	0: Negative sampling is unavailable. Recommended value range: 5 to 10.	0
sample	Optional. The downward sampling threshold.	0 or smaller values: downward sampling is unavailable. Recommended value range: 1e-3 to 1e-5.	0
alpha	Optional. The initial learning rate.	A value greater than 0	0.025
iterTrain	Optional. The number of training iterations.	A value greater than or equal to 1	1

Parameter	Description	Valid values	Default value
randomWindow	Optional. This parameter specifies whether to randomly set the size of the window.	1: The window size is randomly generated. The window size value will range from 1 to 5. 0: The window size is determined by the window parameter.	1
outVocabularyTableName	Optional. The name of the output word list.	Table name	No Output Word List is generated by default.
outVocabularyPartition	Optional. The partition in the output word list.	Partition name	The output word list is non-partitioned by default.
outputTableName	Required. The name of the output table.	Table name	-
outputPartition	Optional. The information about partitions in the output table.	Partition name	The output table is non-partitioned by default.

4.4.10. Network analysis

4.4.10.1. K-core

The k-core of a graph is the largest subgraph in which every vertex is connected to at least k other vertices within the subgraph. The coreness of a vertex is k if it belongs to the k-core but is not included in the (k+1)-core. Therefore, the coreness of a vertex whose degree is 1 must be 0. The graph coreness is equal to that of the vertex with the largest coreness.

Parameter settings

k: Required. The value of the coreness. Default value: 3.

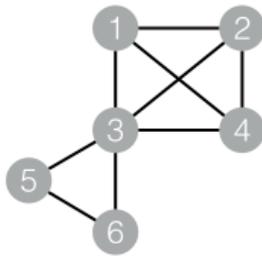
Examples - Testing data

SQL statement to generate data:

```
drop table if exists KCore_func_test_edge;
create table KCore_func_test_edge as
select * from (
select '1' as flow_out_id,
'2' as flow_in_id from dual union all
select '1' as flow_out_id,
'3' as flow_in_id from dual union all
select '1' as flow_out_id,
'4' as flow_in_id from dual union all
select '2' as flow_out_id,
'3' as flow_in_id from dual union all
select '2' as flow_out_id,
'4' as flow_in_id from dual union all
select '3' as flow_out_id,
'4' as flow_in_id from dual union all
select '3' as flow_out_id,
'5' as flow_in_id from dual union all
select '3' as flow_out_id,
'6' as flow_in_id from dual union all
select '5' as flow_out_id,
'6' as flow_in_id from dual )tmp;
```

Graph structure shows the group structure.

Graph structure



Set K to 2. Output shows the output.

Output

node1	node2
1	2
1	3
1	4
2	1
2	3
2	4
3	1
3	2
3	4
4	1
4	2
4	3

PAI command

```

pai -name KCore
-project algo_public
-DinputEdgeTableName=KCore_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=KCore_func_test_result
-Dk=2;
    
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the edge table.	Yes	-
toVertexCol	The end vertex column in the edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64
k	The number of cores.	Yes	3

4.4.10.2. Single-source shortest path

The single-source shortest path (SSSP) refers to the shortest path between a vertex and all other vertices as calculated by the Dijkstra algorithm.

Parameter settings

Start Vertex ID: Required. The ID of the start vertex used to calculate the shortest paths.

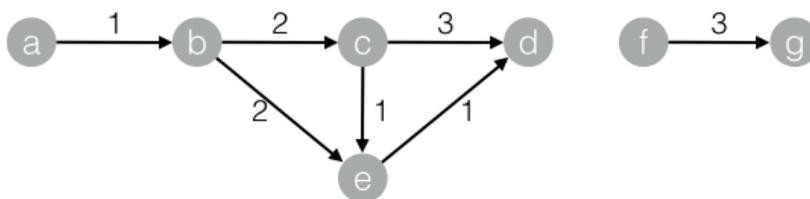
Examples - Testing data

SQL statement to generate data:

```
drop table if exists SSSP_func_test_edge;
create table SSSP_func_test_edge
as select
flow_out_id,flow_in_id,edge_weight from (
select "a" as flow_out_id,
"b" as flow_in_id,
1.0 as edge_weight from dual union all
select "b" as flow_out_id,
"c" as flow_in_id,
2.0 as edge_weight from dual union all
select "c" as flow_out_id,
"d" as flow_in_id,
1.0 as edge_weight from dual union all
select "b" as flow_out_id,
"e" as flow_in_id,
2.0 as edge_weight from dual union all
select "e" as flow_out_id,
"d" as flow_in_id,
1.0 as edge_weight from dual union all
select "c" as flow_out_id,
"e" as flow_in_id,
1.0 as edge_weight from dual union all
select "f" as flow_out_id,
"g" as flow_in_id,
3.0 as edge_weight from dual union all
select "a" as flow_out_id,
"d" as flow_in_id,
4.0 as edge_weight from dual ) tmp ;
```

Graph structure shows the graph structure.

Graph structure



Output

start_node	dest_node	distance	distance_cnt
a	b	1.0	1
a	c	3.0	1
a	d	4.0	3
a	a	0.0	0
a	e	3.0	1

PAI command

```

pai -name SSSP
-project algo_public
-DinputEdgeTableName=SSSP_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=SSSP_func_test_result
-DhasEdgeWeight=true
-DedgeWeightCol=edge_weight
-DstartVertex=a;
  
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64
startVertex	The ID of the start vertex.	Yes	-
hasEdgeWeight	Specifies whether the edges in the input edge table have weights.	No	false
edgeWeightCol	The edge weight column in the input edge table.	No	-

4.4.10.3. PageRank

The PageRank algorithm is used to sort and calculate the rankings of web pages based on their link sources.

Features

The basic principle of the PageRank algorithm is as follows: The more web pages that direct to a web page, the more importance or higher quality the web page has. In addition to the number of links directing to a web page, the weight of the web page and the number of outgoing links are also considered during page ranking. For a social network of users, the edge weight is an important factor in addition to the influence of the users. For example, a Sina Weibo user is more likely to have influence on their family, friends, classmates, and colleagues than they will on followers with a weaker relationship. In the social network, the edge weight is equivalent to the user-to-user relationship strength index. The PageRank formula with connection weight is as follows:

$$W(A) = (1 - d) + d * \left(\sum_i W(i) * C(Ai) \right)$$

In the formula, $W(i)$ represents the weight of node i , $C(A,i)$ represents the link weight, and d represents the damping coefficient. W is the influence index of each user and represents the node weight after the algorithm iteration becomes stable.

Parameter settings

Maximum Iterations: Optional. The number of iterations performed before the algorithm automatically converges. Default value: 30.

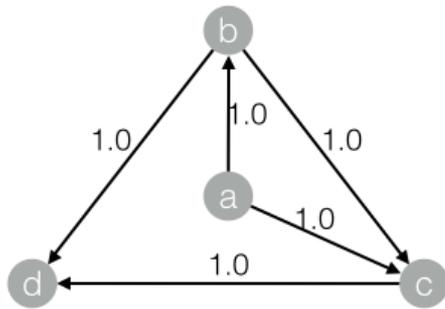
Examples - Testing data

SQL statement to generate data:

```
drop table if exists PageRankWithWeight_func_test_edge;
create table PageRankWithWeight_func_test_edge
as select * from (
select 'a' as flow_out_id,
'b' as flow_in_id,
1.0 as weight from dual union all
select 'a' as flow_out_id,
'c' as flow_in_id,
1.0 as weight from dual union all
select 'b' as flow_out_id,
'c' as flow_in_id,
1.0 as weight from dual union all
select 'b' as flow_out_id,
'd' as flow_in_id,
1.0 as weight from dual union all
select 'c' as flow_out_id,
'd' as flow_in_id,1.0 as weight from dual )tmp ;
```

[Graph structure](#) shows the graph structure.

Graph structure



Output

node	weight
a	0.0375
b	0.06938
c	0.12834
d	0.20556

PAI command

```

pai -name PageRankWithWeight
-project algo_public
-DinputEdgeTableName=PageRankWithWeight_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=PageRankWithWeight_func_test_result
-DhasEdgeWeight=true
-DedgeWeightCol=weight
-DmaxIter 100;
    
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-

Parameter	Description	Required	Default value
<code>outputTablePartitions</code>	The partitions in the output table.	No	-
<code>lifecycle</code>	The lifecycle of the output table.	No	-
<code>workerNum</code>	The number of workers.	No	-
<code>workerMem</code>	The memory size per worker.	No	4096
<code>splitSize</code>	The data split size.	No	64
<code>hasEdgeWeight</code>	Specifies whether the edges in the input edge table have weights.	No	false
<code>edgeWeightCol</code>	The edge weight column in the input edge table.	No	-
<code>maxIter</code>	The maximum number of iterations.	No	30

4.4.10.4. Label propagation clustering

Graph clustering is used to divide a graph into subgraphs based on the topology of the graph so that the links between the nodes in a subgraph are more than the links between the subgraphs. The label propagation algorithm (LPA) is a graph-based semi-supervised machine learning algorithm. The labels of a node (community) depend on those of the neighboring nodes. The degree of dependence is determined by the similarity between nodes. Data becomes stable by iterative propagation update.

Parameters

Maximum Iterations: Optional. The maximum number of iterations. Default value: 30.

Examples - Testing data

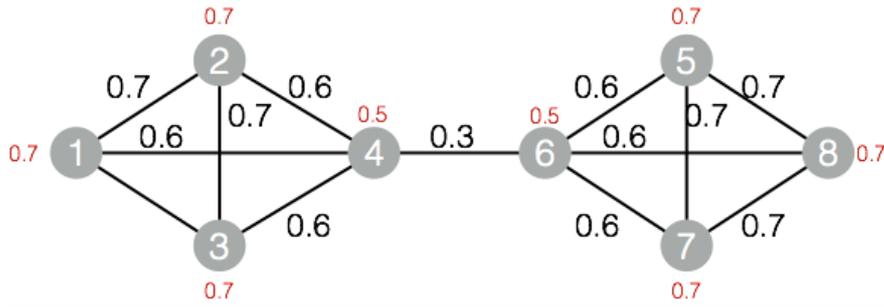
SQL statement to generate data:

```
drop table if exists LabelPropagationClustering_func_test_edge;
create table LabelPropagationClustering_func_test_edge
as select * from (
select '1' as flow_out_id,
'2' as flow_in_id,
0.7 as edge_weight from dual union all
select '1' as flow_out_id,
'3' as flow_in_id,
0.7 as edge_weight from dual union all
select '1' as flow_out_id,
'4' as flow_in_id,
0.6 as edge_weight from dual union all
select '2' as flow_out_id,
'3' as flow_in_id,
0.7 as edge_weight from dual union all
select '2' as flow_out_id,
```

```
'4' as flow_in_id,
0.6 as edge_weight from dual union all
select '3' as flow_out_id,
'4' as flow_in_id,
0.6 as edge_weight from dual union all
select '4' as flow_out_id,
'6' as flow_in_id,
0.3 as edge_weight from dual union all
select '5' as flow_out_id,
'6' as flow_in_id,
0.6 as edge_weight from dual union all
select '5' as flow_out_id,
'7' as flow_in_id,
0.7 as edge_weight from dual union all
select '5' as flow_out_id,
'8' as flow_in_id,
0.7 as edge_weight from dual union all
select '6' as flow_out_id,
'7' as flow_in_id,
0.6 as edge_weight from dual union all
select '6' as flow_out_id,
'8' as flow_in_id,
0.6 as edge_weight from dual union all
select '7' as flow_out_id,
'8' as flow_in_id,
0.7 as edge_weight from dual )tmp ;
drop table if exists LabelPropagationClustering_func_test_node;
create table LabelPropagationClustering_func_test_node
as select * from (
select '1' as node,
0.7 as node_weight from dual union all
select '2' as node,
0.7 as node_weight from dual union all
select '3' as node,
0.7 as node_weight from dual union all
select '4' as node,
0.5 as node_weight from dual union all
select '5' as node,
0.7 as node_weight from dual union all
select '6' as node,
0.5 as node_weight from dual union all
select '7' as node,
0.7 as node_weight from dual union all
select '8' as node,
0.7 as node_weight from dual )tmp ;
```

Group structure shows the group structure.

Group structure



Output

node	group_id
1	1
2	1
3	1
4	1
5	5
6	5
7	5
8	5

PAI command

```

pai -name LabelPropagationClustering
-project algo_public
-DinputEdgeTableName=LabelPropagationClustering_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DinputVertexTableName=LabelPropagationClustering_func_test_node
-DvertexCol=node
-DoutputTableName=LabelPropagationClustering_func_test_result
-DhasEdgeWeight=true
-DedgeWeightCol=edge_weight
-DhasVertexWeight=true
-DvertexWeightCol=node_weight
-DrandSelect=true
-DmaxIter=100;
    
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.

Parameter	Description	Required	Default value
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
inputVertexTableName	The name of the input vertex table.	Yes	-
inputVertexTablePartitions	The partitions in the input vertex table.	No	The whole table is selected by default.
vertexCol	The vertex column in the input vertex table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64
hasEdgeWeight	Specifies whether the edges in the input edge table have weights.	No	false
edgeWeightCol	The edge weight column in the input edge table.	No	-
hasVertexWeight	Specifies whether the vertices in the input vertex table have weights.	No	false
vertexWeightCol	The vertex weight column in the input vertex table.	No	-
randSelect	Specifies whether the maximum label value is to be randomly selected.	No	false
maxIter	The maximum number of iterations.	No	30

4.4.10.5. Label propagation classification

Label propagation classification is a semi-supervised classification algorithm. It uses the label information of labeled nodes to predict the label information for unlabeled nodes.

Features

During algorithm execution, the labels of each node are propagated to the neighboring nodes based on the similarity between the nodes. In each step of propagation, a node updates its labels based on the labels of the neighboring nodes so that the node is more similar to the neighboring nodes. The higher the similarity, the more labeling influences the neighboring nodes have on that node, and the easier it is for the labels to be propagated. During label propagation, the labels of the labeled data remain unchanged. These labels serve as sources for propagation to the unlabeled data.

After the iterations end, the probability distributions of similar nodes tend to be similar. These nodes can be classified into the same category. This completes the label propagation.

Parameter settings

Damping factor: The default value is 0.8. Convergence factor: The default value is 0.000001.

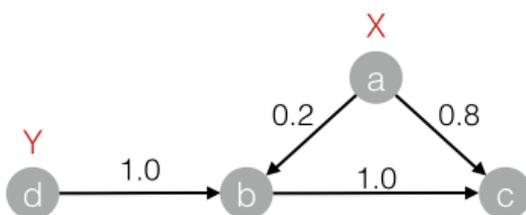
Examples - Testing data

SQL statement to generate data:

```
drop table if exists LabelPropagationClassification_func_test_edge;
create table LabelPropagationClassification_func_test_edge
as select * from (
select 'a' as flow_out_id,
'b' as flow_in_id,
0.2 as edge_weight from dual union all
select 'a' as flow_out_id,
'c' as flow_in_id,
0.8 as edge_weight from dual union all
select 'b' as flow_out_id,
'c' as flow_in_id,
1.0 as edge_weight from dual union all
select 'd' as flow_out_id,
'b' as flow_in_id,
1.0 as edge_weight from dual )tmp ;
drop table if exists LabelPropagationClassification_func_test_node;
create table LabelPropagationClassification_func_test_node
as select * from (
select 'a' as node,
'X' as label,
1.0 as label_weight from dual union all
select 'd' as node,
'Y' as label,
1.0 as label_weight from dual )tmp ;
```

Graph structure shows the graph structure.

Graph structure



Output

node	tag	weight
a	X	1.0
b	X	0.16667
b	Y	0.83333
c	X	0.53704
c	Y	0.46296
d	Y	1.0

PAI command

```

pai -name LabelPropagationClassification
  -project algo_public
  -DinputEdgeTableName=LabelPropagationClassification_func_test_edge
  -DfromVertexCol=flow_out_id
  -DtoVertexCol=flow_in_id
  -DinputVertexTableName=LabelPropagationClassification_func_test_node
  -DvertexCol=node
  -DvertexLabelCol=label
  -DoutputTableName=LabelPropagationClassification_func_test_result
  -DhasEdgeWeight=true
  -DedgeWeightCol=edge_weight
  -DhasVertexWeight=true
  -DvertexWeightCol=label_weight
  -Dalpha=0.8
  -Depsilon=0.000001;

```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
inputVertexTableName	The name of the input vertex table.	Yes	-
inputVertexTablePartitions	The partitions in the input vertex table.	No	The whole table is selected by default.

Parameter	Description	Required	Default value
vertexCol	The vertex column in the input vertex table.	Yes	-
vertexLabelCol	The vertex label column in the input vertex table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64
hasEdgeWeight	Specifies whether the edges in the input edge table have weights.	No	false
edgeWeightCol	The edge weight column in the input edge table.	No	-
hasVertexWeight	Specifies whether the vertices in the input vertex table have weights.	No	false
vertexWeightCol	The vertex weight column in the input vertex table.	No	-
alpha	The damping coefficient.	No	0.8
epsilon	The convergence coefficient.	No	0.000001
maxIter	The maximum number of iterations.	No	30

4.4.10.6. Modularity

Modularity is used to measure the structure of the community network. It measures the closeness of the communities divided from a network structure. A value larger than 0.3 represents an obvious community structure.

Examples - Testing data

For more information, see [Label propagation clustering](#).

Output

val
0.4230769

PAI command

```

pai -name Modularity
-project algo_public
-DinputEdgeTableName=Modularity_func_test_edge
-DfromVertexCol=flow_out_id
-DfromGroupCol=group_out_id
-DtoVertexCol=flow_in_id
-DtoGroupCol=group_in_id
-DoutputTableName=Modularity_func_test_result;
  
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
fromGroupCol	The start vertex group in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
toGroupCol	The end vertex group in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096

Parameter	Description	Required	Default value
splitSize	The data split size.	No	64

4.4.10.7. Maximum connected subgraph

In an undirected graph G, vertex A is connected to vertex B if a path exists between the two vertices. Graph G contains several subgraphs. Each vertex is connected to other vertices in the same subgraph. Vertices in different subgraphs are not connected. In this case, the subgraphs of graph G are called maximum connected subgraphs.

Features

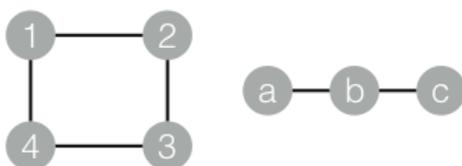
Examples - Testing data

SQL statement to generate data:

```
drop table if exists MaximalConnectedComponent_func_test_edge;
create table MaximalConnectedComponent_func_test_edge
as select * from (
select '1' as flow_out_id,
'2' as flow_in_id from dual union all
select '2' as flow_out_id,
'3' as flow_in_id from dual union all
select '3' as flow_out_id,
'4' as flow_in_id from dual union all
select '1' as flow_out_id,
'4' as flow_in_id from dual union all
select 'a' as flow_out_id,
'b' as flow_in_id from dual union all
select 'b' as flow_out_id,
'c' as flow_in_id from dual )tmp;
drop table if exists MaximalConnectedComponent_func_test_result;
create table MaximalConnectedComponent_func_test_result ( node string, grp_id string);
```

Graph structure shows the graph structure.

Graph structure



Output

node	grp_id
1	4
2	4
3	4
4	4
a	c
b	c
c	c

PAI command

```
pai -name MaximalConnectedComponent
-project algo_public
-DinputEdgeTableName=MaximalConnectedComponent_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=MaximalConnectedComponent_func_test_result;
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64

4.4.10.8. Vertex clustering coefficient

This coefficient is used to calculate the peripheral density of a vertex in an undirected graph G. The density of a star network is 0, and that of a fully meshed network is 1.

Parameter settings

maxEdgeCnt: Optional. If the node degree is larger than the value of this parameter, sampling is required.
Default value: 500.

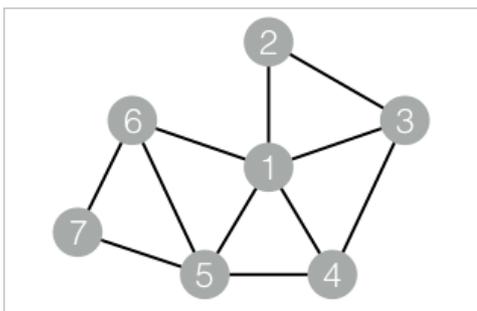
Examples - Testing data

SQL statement to generate data:

```
drop table if exists NodeDensity_func_test_edge;
create table NodeDensity_func_test_edge
as select * from (
select '1' as flow_out_id,
'2' as flow_in_id from dual union all
select '1' as flow_out_id,
'3' as flow_in_id from dual union all
select '1' as flow_out_id,
'4' as flow_in_id from dual union all
select '1' as flow_out_id,
'5' as flow_in_id from dual union all
select '1' as flow_out_id,
'6' as flow_in_id from dual union all
select '2' as flow_out_id,
'3' as flow_in_id from dual union all
select '3' as flow_out_id,
'4' as flow_in_id from dual union all
select '4' as flow_out_id,
'5' as flow_in_id from dual union all
select '5' as flow_out_id,
'6' as flow_in_id from dual union all
select '5' as flow_out_id,
'7' as flow_in_id from dual union all
select '6' as flow_out_id,
'7' as flow_in_id from dual )tmp;
drop table if exists NodeDensity_func_test_result;
create table NodeDensity_func_test_result ( node string, node_cnt bigint, edge_cnt bigint, density double, log_density double);
```

Graph structure shows the graph structure.

Graph structure



Output

```
1,5,4,0.4,1.45657
2,2,1,1.0,1.24696
3,3,2,0.66667,1.35204
4,3,2,0.66667,1.35204
5,4,3,0.5,1.41189
6,3,2,0.66667,1.35204
7,2,1,1.0,1.24696
```

PAI command

```
pai -name NodeDensity
-project algo_public
-DinputEdgeTableName=NodeDensity_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=NodeDensity_func_test_result
-DmaxEdgeCnt=500;
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
maxEdgeCnt	If the node degree is larger than the value of this parameter, sampling is required.	No	500
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096

Parameter	Description	Required	Default value
splitSize	The data split size.	No	64

4.4.10.9. Edge clustering coefficient

This coefficient is used to calculate the peripheral density of each edge in an undirected graph G.

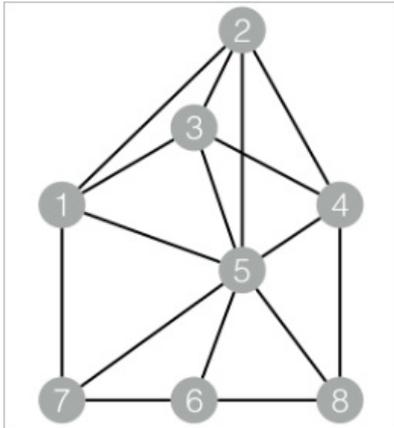
Examples - Testing data

SQL statement to generate data:

```
drop table if exists EdgeDensity_func_test_edge;
create table EdgeDensity_func_test_edge
as select * from (
select '1' as flow_out_id,
'2' as flow_in_id from dual union all
select '1' as flow_out_id,
'3' as flow_in_id from dual union all
select '1' as flow_out_id,
'5' as flow_in_id from dual union all
select '1' as flow_out_id,
'7' as flow_in_id from dual union all
select '2' as flow_out_id,
'5' as flow_in_id from dual union all
select '2' as flow_out_id,
'4' as flow_in_id from dual union all
select '2' as flow_out_id,
'3' as flow_in_id from dual union all
select '3' as flow_out_id,
'5' as flow_in_id from dual union all
select '3' as flow_out_id,
'4' as flow_in_id from dual union all
select '4' as flow_out_id,
'5' as flow_in_id from dual union all
select '4' as flow_out_id,
'8' as flow_in_id from dual union all
select '5' as flow_out_id,
'6' as flow_in_id from dual union all
select '5' as flow_out_id,
'7' as flow_in_id from dual union all
select '5' as flow_out_id,
'8' as flow_in_id from dual union all
select '7' as flow_out_id,
'6' as flow_in_id from dual union all
select '6' as flow_out_id,
'8' as flow_in_id from dual )tmp;
drop table if exists EdgeDensity_func_test_result;
create table EdgeDensity_func_test_result ( node1 string, node2 string, node1_edge_cnt bigint, node2_edge_c
nt bigint, triangle_cnt bigint, density double );
```

[Graph structure](#) shows the graph structure.

Graph structure



Output

```
1,2,4,4,2,0.5
2,3,4,4,3,0.75
2,5,4,7,3,0.75
3,1,4,4,2,0.5
3,4,4,4,2,0.5
4,2,4,4,2,0.5
4,5,4,7,3,0.75
5,1,7,4,3,0.75
5,3,7,4,3,0.75
5,6,7,3,2,0.66667
5,8,7,3,2,0.66667
6,7,3,3,1,0.33333
7,1,3,4,1,0.33333
7,5,3,7,2,0.66667
8,4,3,4,1,0.33333
8,6,3,3,1,0.33333
```

PAI command

```
pai -name EdgeDensity
-project algo_public
-DinputEdgeTableName=EdgeDensity_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=EdgeDensity_func_test_result;
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-

Parameter	Description	Required	Default value
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64

4.4.10.10. Counting triangle

All triangles can be output to an undirected graph G.

Parameter settings

maxEdgeCnt: Optional. If the node degree is larger than the value of this parameter, sampling is required.
 Default value: 500.

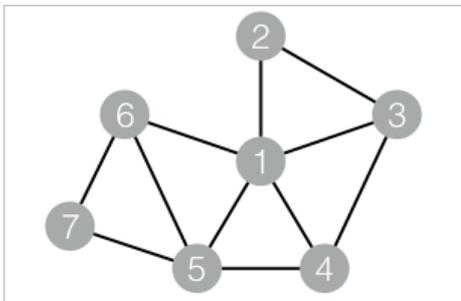
Examples - Testing data

SQL statement to generate data:

```
drop table if exists TriangleCount_func_test_edge;
create table TriangleCount_func_test_edge
as select * from (
select '1' as flow_out_id,
'2' as flow_in_id from dual union all
select '1' as flow_out_id,
'3' as flow_in_id from dual union all
select '1' as flow_out_id,
'4' as flow_in_id from dual union all
select '1' as flow_out_id,
'5' as flow_in_id from dual union all
select '1' as flow_out_id,
'6' as flow_in_id from dual union all
select '2' as flow_out_id,
'3' as flow_in_id from dual union all
select '3' as flow_out_id,
'4' as flow_in_id from dual union all
select '4' as flow_out_id,
'5' as flow_in_id from dual union all
select '5' as flow_out_id,
'6' as flow_in_id from dual union all
select '5' as flow_out_id,
'7' as flow_in_id from dual union all
select '6' as flow_out_id,
'7' as flow_in_id from dual )tmp;
drop table if exists TriangleCount_func_test_result;
create table TriangleCount_func_test_result ( node1 string, node2 string, node3 string);
```

Graph structure shows the graph structure.

Graph structure



Output

```
1,2,3
1,3,4
1,4,5
1,5,6
5,6,7
```

PAI command

```

pai -name TriangleCount
-project algo_public
-DinputEdgeTableName=TriangleCount_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=TriangleCount_func_test_result;
    
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
maxEdgeCnt	If the node degree is larger than the value of this parameter, sampling is required.	No	500
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64

4.4.10.11. Tree depth

In a tree network, this component outputs the depth of each node in a tree and the tree ID.

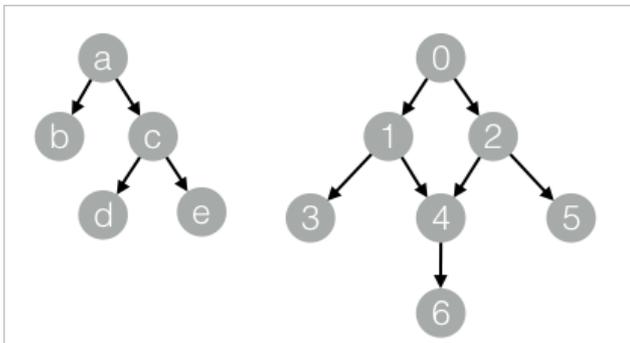
Examples - Testing data

SQL statement to generate data:

```
drop table if exists TreeDepth_func_test_edge;
create table TreeDepth_func_test_edge
as select * from (
select '0' as flow_out_id,
'1' as flow_in_id from dual union all
select '0' as flow_out_id,
'2' as flow_in_id from dual union all
select '1' as flow_out_id,
'3' as flow_in_id from dual union all
select '1' as flow_out_id,
'4' as flow_in_id from dual union all
select '2' as flow_out_id,
'4' as flow_in_id from dual union all
select '2' as flow_out_id,
'5' as flow_in_id from dual union all
select '4' as flow_out_id,
'6' as flow_in_id from dual union all
select 'a' as flow_out_id,
'b' as flow_in_id from dual union all
select 'a' as flow_out_id,
'c' as flow_in_id from dual union all
select 'c' as flow_out_id,
'd' as flow_in_id from dual union all
select 'c' as flow_out_id,
'e' as flow_in_id from dual )tmp;
drop table if exists TreeDepth_func_test_result;
create table TreeDepth_func_test_result ( node string, root string, depth bigint );
```

Graph structure shows the graph structure.

Graph structure



Output

```
0,0,0
1,0,1
2,0,1
3,0,2
4,0,2
5,0,2
6,0,3
a,a,0
b,a,1
c,a,1
d,a,2
e,a,2
```

PAI command

```
pai -name TreeDepth
-project algo_public
-DinputEdgeTableName=TreeDepth_func_test_edge
-DfromVertexCol=flow_out_id
-DtoVertexCol=flow_in_id
-DoutputTableName=TreeDepth_func_test_result;
```

Algorithm parameters

Parameters

Parameter	Description	Required	Default value
inputEdgeTableName	The name of the input edge table.	Yes	-
inputEdgeTablePartitions	The partitions selected from the input edge table.	No	The whole table is selected by default.
fromVertexCol	The start vertex column in the input edge table.	Yes	-
toVertexCol	The end vertex column in the input edge table.	Yes	-
outputTableName	The name of the output table.	Yes	-
outputTablePartitions	The partitions in the output table.	No	-
lifecycle	The lifecycle of the output table.	No	-
workerNum	The number of workers.	No	-
workerMem	The memory size per worker.	No	4096
splitSize	The data split size.	No	64

4.4.11. Tools

4.4.11.1. SQL script

You can use the SQL script editor to write SQL statements.

1. Drag and drop the **SQL Script** component onto the canvas.
2. Connect the input table to the **SQL Script** component, and then click **SQL Script**. The following configuration pane is displayed.
3. Write an SQL script in the text box.
 - An SQL script supports one to four inputs and one output.

- You can write only one SQL statement.
- The input data is automatically mapped to tables t1 through t4. You can directly call `#{t1}`, `#{t2}`, `#{t3}`, and `#{t4}` without specifying the table names.
- The sample SQL script calculates the number of rows in the input table.

4.4.12. Financials

4.4.12.1. Binning

The Binning component performs data binning based on equal-width or equal-frequency.

PAI command

```
PAI -name binning  
-project algo_public  
-DinputTableName=input  
-DoutputTableName=output
```

Parameters

Parameter	Description	Valid value	Default
inputTableName	Required. The name of the input table.	Table name	N/A
outputTableName	Required. The name of the output table.	Table name	N/A
selectedColNames	Optional. The names of columns selected from the input table for data binning.	Column name	All columns are selected, except for the label column.
labelColumn	Optional. The name of the column that stores the labels.	Column name	No label column is selected.
validTableName	Required when the binning mode (binningMethod) is set to auto. The name of the table used for calibration.	Table name	No table is specified for calibration.
validTablePartitions	Optional. The partitions selected from the calibration table.	Partition name	The whole table is selected.
inputTablePartitions	Optional. The partitions selected from the input table.	Partition name	The whole table is selected.
inputBinTableName	Optional. The name of the input binning table.	Table name	No binning table is specified.

Parameter	Description	Valid value	Default
selectedBinColNames	Optional. The names of the columns selected from the binning table.	Column name	No column is selected.
positiveLabel	Optional. The value used to represent positive samples.	N/A	1
nDivide	Optional. The number of bins.	A positive integer	10
colsNDivide	Optional. The numbers of bins customized for specified columns. Example: col0:3,col2:5. Columns specified in the colsNDivide parameter but not included in the selectedColNames parameter are also processed by the Binning component. For example, if selectedColNames is set to col0,col1 and colsNDivide is set to col0:3,col2:5, data binning is performed based on col0:3,col1:10,col2:5.	N/A	No custom binning rule is specified.
isLeftOpen	Optional. The type of the interval, which can be left-open, right-closed or left-closed, right-open.	true and false	true
stringThreshold	Optional. The discrete value threshold. Values below this threshold are put into other bins.	N/A	No discrete value threshold is set.
colsStringThreshold	Optional. The thresholds for specified columns. Specify the values in the same format of the colsNDivide parameter.	N/A	No threshold is set.
binningMethod	Optional. The binning mode.	quantile (equal-frequency), bucket (equal-width), and auto (automatic binning). If you select auto, monotonic binning is used based on equal-frequency.	quantile
lifecycle	Optional. The lifecycle of the output table.	A positive integer	No lifecycle is set.

Parameter	Description	Valid value	Default
coreNum	Optional. The number of cores.	A positive integer	Automatically calculated.
memSizePerCore	Optional. The memory size per core.	A positive integer	Automatically calculated.

Constraints

Binning constraints must be used together with the Scorecard Training component. You can add constraints to the weight of each dummy variable when the Binning component discretizes features and transforms them into dummy variables. The definitions of the constraints are as follows:

- Ascending order: adds weights to the dummy variables of a feature based on the index values in ascending order. This means that dummy variables with higher index values have larger weights.
- Descending order: adds weights to the dummy variables of a feature based on the index values in descending order. This means that dummy variables with lower index values have lighter weights.
- Same weight: the weights of two dummy variables of a feature must be the same.
- Set weight to 0: sets the weight of a dummy variable to 0.
- Set weight to a specified value: sets the weight of a dummy variable to a floating point value.
- WOE order: adds weights to the dummy variables of a feature based on the WOE values in ascending order. This means that dummy variables with higher WOE values have larger weights.

4.4.12.2. Data conversion

Parameters

Parameter	Description
inputFeatureTableName	Required. The name of the input feature table.
inputBinTableName	Required. The name of the binning result table.
inputFeatureTablePartitions	Optional. The partitions selected from the feature table. By default, all partitions are selected.
outputTableName	Required. The name of the output table.
featureColNames	Optional. The names of the features selected from the input feature table. By default, all columns are selected.
metaColNames	Optional. The names of the columns to be reserved in the output table without data conversion. By default, no column is reserved. You can specify columns such as label and sample_id.
transformType	Optional. The type of data conversion. Valid values: normalize (normalization), dummy (discretization), and woe (WOE transformation). The default is dummy.
itemDelimiter	Optional. The delimiter used to separate features. By default, commas (,) are used. Only data discretization supports this parameter.

Parameter	Description
kvDelimiter	Optional. The delimiter used to separate keys and values. By default, colons (:) are used. Only data discretization supports this parameter.
lifecycle	Optional. The lifecycle of the output table. By default, no lifecycle is set.
coreNum	Optional. The number of cores. By default, the number of cores is automatically calculated.
memSizePerCore	Optional. The memory size per core in megabytes. By default, the memory size per core is automatically calculated.

instance

PAI command

```
PAI -name data_transform
  -project algo_public
  -DinputFeatureTableName=feature_table
  -DinputBinTableName=bin_table
  -DoutputTableName=output_table
  -DmetaColNames=label
  -DfeatureColNames=feaname1,feaname2
```

Normalization

The Normalization component transforms variable values to a scale of 0 to 1. Missing values are imputed with zeros. The algorithm is as follows:

```
if feature_raw_value == null or feature_raw_value == 0 then
  feature_norm_value = 0.0
else
  bin_index = FindBin(bin_table, feature_raw_value)
  bin_width = round(1.0 / bin_count * 1000) / 1000.0
  feature_norm_value = 1.0 - (bin_count - bin_index - 1) * bin_width
```

Output format

Normalization and WOE conversion tables output regular tables.

Dummy variable conversion in discretization outputs a table that contains KV pairs. The output variables are in the format of `[${feaname}]_bin_${bin_id}`. Taking variable `sns` as an example:

- If the variable `sns` is put into the second bin, the output variable is `[sns]_bin_2`.
- If the variable `sns` does not have a value, it is put into an empty bin. The output variable is `[sns]_bin_null`.
- If the variable `sns` has a value but it cannot be put into any of the predefined bins, it is put into the else bin. The output variable is `[sns]_bin_else`.

4.4.12.3. Scorecard training

The scorecard is a modeling tool widely used for credit score evaluation. It uses binning to discretize variables, and then uses linear models (logistic regression or linear regression) to train a model. The model training process includes feature selection and score transformation. The scorecard also allows you to add constraints to the variables during model training.

Note If you use the scorecard without binning, the entire model training process is equivalent to logistic regression or linear regression.

Feature engineering

The main difference between the scorecard and normal linear models is that the scorecard performs feature engineering before it trains linear models. The Scorecard component provides two methods for feature engineering. You must use binning to discretize the features first no matter which method you choose. One of the methods is to use one hot encoding to encode the variable binning results, and then generate N dummy variables. N represents the number of bins. The other method is to use Weight of Evidence (WOE) transformation. It replaces the original value of a variable with the WOE value of the bin where the variable is placed.

Note You can add constraints for each variable when you transform variables to dummy variables.

Score transformation

In the credit score evaluation scenario, you must use linear transformation to transform the odds of the samples in the prediction results to credit scores as follows:

$$\log(\text{odds}) = \sum(wx) = a \text{ scaled_score} + b$$

The parameters in the formula are as follows:

- scaledValue: specifies a base point to be scaled.
- odds: specifies an odd level.
- pdo: specifies the points to double the odds.

For example, if the scaledValue, odds, and pdo parameters are set to 800, 50, and 25, the two vectors that determine the score scale are as follows:

$$\begin{aligned}\log(40) &= a * 800 + b \\ \log(80) &= a * 825 + b\end{aligned}$$

Calculate the values of a and b, and then perform a linear transformation to obtain the scores.

The scaling information is specified in JSON format by using the -Dscale parameter as follows:

```
{"scaledValue":800,"odds":50,"pdo":25}
```

The three parameters must be set at the same time.

Add constraints

You can add constraints for variables during scorecard training. You can set the score of a bin to a fixed value, set a proportion between the scores of two bins, or limit the scores of bins. For example, you can sort the scores of the bins by WOE. The implementation of constraints depends on the underlying optimization algorithms with constraints. You can set a constraint in the Binning component. After you set the parameters, the component generates a constraint in JSON format, and then passes the constraint to the component connected to it. The supported constraints are as follows:

- <: sorts the weights of the variables in ascending order.
- >: sorts the weights of the variables in descending order.
- =: sets the weights of the variables to a fixed value.
- %: sets a proportion between the weights of two variables.
- UP: sets an upper limit for the weights of the variables.
- LO: sets a lower limit for the weights of the variables.

A constraint is stored in a table as a JSON string. The table contains only one row and one column.

```
{
  "name": "feature0",
  "<": [
    [0,1,2,3]
  ],
  ">": [
    [4,5,6]
  ],
  "=": [
    "3:0","4:0.25"
  ],
  "%": [
    ["6:1.0","7:1.0"]
  ]
}
```

Built-in constraints

Every original variable has a default constraint. The average score of the population in a variable must be 0. Based on this constraint, the scaled_weight in the intercept options of the scorecard model equals the average score of the entire population.

Optimization algorithms

In the advanced options of the Scorecard component, you can select optimization algorithms to be used in model training. The supported optimization algorithms are as follows:

- L-BFGS
- Newton's Method
- Barrier Method
- SQP

L-BFGS is a first-order optimization algorithm for processing large amounts of feature data. Newton's Method is a classic tier-2 optimization algorithm. It is fast in regression and accurate. However, it is not suitable for processing large amounts of feature data because it needs to calculate the second-order Hessian Matrix. The two algorithms do not have any constraints. When these algorithms are selected, the system automatically ignores the constraints.

If you do not want the system to ignore the constraints, select Barrier Method or SQP. Barrier Method and SQP are second-order optimization algorithms. When no constraint is set, they are equivalent to Newton's Method. Barrier Method and SQP have minor differences in performance and accuracy. We recommend that you choose SQP. If you are not familiar with optimization algorithms, we recommend that you choose the Auto-selected by default option. The system will automatically select an optimization algorithm based on the amount of the data and the constraints.

Feature selection

The Scorecard component supports stepwise feature selection. Stepwise is a combination of forward selection and backward selection. Each time the system selects a new variable by forward selection and adds it to the model, it must perform a backward selection. The backward selection starts with the variables in the model, and eliminates the ones with significance not meeting the requirements. Stepwise feature selection supports various types of target functions and feature transformation methods. Therefore, stepwise feature selection also supports multiple selection standards. Currently, the following standards are supported:

- Marginal contribution: It can be applied to all target functions and feature engineering methods.
- Score test: It only supports WOE transformation and logistic regression without feature engineering.
- F test: It only supports WOE transformation and linear regression without feature engineering.

Marginal contribution

The marginal contribution is the difference between the target functions of Model A without Variable X and target functions of Model B with Variable X after both models are trained. It is the marginal contribution of Variable X to all the other variables in Model B. In the scenario of transforming variables to dummy variables by feature engineering, the marginal contribution of Variable X is the difference between the target functions of all dummy variables in Model A without Variable X and target functions of all dummy variables in Model B with Variable X. Therefore, using marginal contribution to select features is supported by all feature engineering methods.

Marginal contribution makes feature selection more open-ended. It is not restricted to a certain type of models. Only variables that contribute to the target functions are passed to the model. Marginal contribution has certain disadvantages when compared with statistical significance. Typically, statistical significance chooses 0.05 as its threshold. Marginal contribution does not provide a recommended threshold for beginners. We recommend that you set the threshold to $10E-5$.

Score test

Score test is only suitable for feature selection in logistic regression. During a forward selection, a model with only intercept options is trained first. In each subsequent regression, the score chi-squares of the variables that have not been passed to the model are calculated. The variable with the largest score chi-square is passed to the model. The P-value corresponds to the largest score chi-square is also calculated based on chi-square distribution. If the P-value is greater than the given SLENTRY value, feature selection is complete.

After the forward selection is complete, a backward selection is performed for the variable passed to the model. The Wald chi-square of the variable and the corresponding P-value are calculated. If the P-value is greater than the given SLSTAY value, the variable is removed from the model. The system then starts a new regression.

F test

F test is only suitable for feature selection in linear regression. During a forward selection, a variable with only intercept options is trained first. In each subsequent regression, the F-values of the variables that have not been passed to the model are calculated. F-value calculation is similar to marginal contribution calculation. Both of them need to train two models to calculate the F-value of a variable. The F-value fits the F distribution. The corresponding P-value can be calculated based on the probability density function of the F distribution. If the P-value is greater than the given SLENTRY value, the variable is not passed to the model, and the forward selection is complete.

The backward selection process uses the F-value to calculate the significance of the variable in a way similar to a score test.

Forcibly selected variables

Before you perform feature selection, you can specify variables to be forcibly passed to the model. The specified variables are passed to the model regardless of their significance. No forward selection or backward selection is performed for these variables.

The number of regressions and significance levels (SLENTY and SLSTAY) are defined by using a JSON string in the -Dselected parameter as follows:

```
{"max_step":2, "slentry": 0.0001, "slstay": 0.0001}
```

If the -Dselected parameter is left empty or max_step is set to 0, no feature selection is performed.

Model report

The Scorecard component outputs data to a model report. The model report contains basic model evaluation statistics, such as the binning information, binning constraints, WOE values, and marginal contribution information. The following table lists the fields contained in the model report:

Field	Type	Description
feaname	string	The name of the feature.
binid	bigint	The ID of a bin.
bin	string	The description of the bin, which indicates the interval of the bin.
constraint	string	The constraints of the bin specified for model training.
weight	double	The weight of a binning variable. For a non-scorecard model without binning, this field indicates the weight of a model variable.
scaled_weight	double	For score transformation in scorecard training, this field indicates the score linearly transformed from the weight of a binning variable.
woe	double	A statistical indicator. It indicates the WOE value of a bin in the training set.
contribution	double	A statistical indicator. It indicates the marginal contribution value of a bin in the training set.
total	bigint	A statistical indicator. It indicates the total number of samples in a bin in the training set.
positive	bigint	A statistical indicator. It indicates the number of positive samples in a bin in the training set.
negative	bigint	A statistical indicator. It indicates the number of negative samples in a bin in the training set.

Field	Type	Description
percentage_pos	double	A statistical indicator. It indicates the proportion between positive samples in a bin and total positive samples in the training set.
percentage_neg	double	A statistical indicator. It indicates the proportion between negative samples in a bin and total negative samples in the training set.
test_woe	double	A statistical indicator. It indicates the WOE value of a bin in the testing set.
test_contribution	double	A statistical indicator. It indicates the marginal contribution value of a bin in the testing set.
test_total	bigint	A statistical indicator. It indicates the total number of samples in a bin in the testing set.
test_positive	bigint	A statistical indicator. It indicates the number of positive samples in a bin in the testing set.
test_negative	bigint	A statistical indicator. It indicates the number of negative samples in a bin in the testing set.
test_percentage_pos	double	A statistical indicator. It indicates the proportion between positive samples in a bin and total positive samples in the testing set.
test_percentage_neg	double	A statistical indicator. It indicates the proportion between negative samples in a bin and total negative samples in the testing set.

Algorithm parameters

Parameter	Description	Valid value	Default
inputTableName	Required. The input table that contains features.	N/A	N/A
inputTablePartitions	Optional. The partitions selected from the input table.	N/A	The whole table is selected.

Parameter	Description	Valid value	Default
inputBinTableName	Optional. The binning result table. If this parameter is set, the system automatically discretizes the original features based on the binning rules in the binning result table.	N/A	N/A
featureColNames	Optional. The names of feature columns selected from the input table.	N/A	All columns in the input table are selected by default, except for the label column.
labelColName	Required. The names of the target columns.	N/A	N/A
outputTableName	Required. The name of the output table.	N/A	N/A
inputConstraintTableName	Optional. A constraint. The constraint is a JSON string stored in a cell of a table.	N/A	
optimization	Optional. The optimization algorithm.	lbfgs, newton, barrier_method, sqp, and auto. Currently, only the sqp and barrier_method algorithms support constraints. If you set the value to auto, the system automatically selects an optimization algorithm based on the input data and corresponding parameters. We recommend that you set the value to auto if you are unfamiliar with the listed optimization algorithms.	auto
loss	Optional. The type of the loss function.	logistic_regression and least_square.	logistic_regression
iterations	Optional. The maximum number of regressions.	N/A	100
l1Weight	Optional. The weight of the L1 regularization parameter. Currently, only lbfgs supports l1weight.	N/A	0

Parameter	Description	Valid value	Default
l2Weight	Optional. The weight of the L2 regularization parameter.	N/A	0
m	Optional. The number of regressions performed by L-BFGS. Only L-BFGS supports this parameter.	N/A	10
scale	Optional. The weight scaling information of the scorecard.	N/A	Null
selected	Optional. Feature selection in scorecard training.	N/A	Null
convergenceTolerance	Optional. The convergence tolerance.	N/A	1e-6
positiveLabel	Optional. The category of positive samples.	N/A	1
lifecycle	Optional. The lifecycle of the output table.	N/A	No lifecycle is set.
coreNum	Optional. The number of vCores.	N/A	Automatically calculated.
memSizePerCore	Optional. The memory size per core.	N/A	Automatically calculated.

4.4.12.4. Scorecard prediction

The Scorecard Prediction component predicts credit scores based on the input data. It uses a model generated by a model training component. Supported model training components include the Scorecard Training, Logistic Regression for Binary Classification (in the Financials folder), and Linear Regression (in the Financials folder) components.

Input parameters

The Scorecard Prediction component has the following parameters:

- **Feature Column:** specifies the feature columns to be used for predicting credit scores. By default, all columns are selected.
- **Columns Reserved in Result Table:** specifies the columns to be appended to the prediction result table without any changes, such as the ID column and target column.
- **Output Variable Score:** specifies whether to output the score of each variable. The final score equals the score in the intercept option plus the scores of all variables.

Score table

The following is an example of the score table output by the component.

The first column churn is the column appended to the result table from the input table. The data in this column does not affect the prediction results. The remaining three columns display the prediction results. The definitions of these columns are as follows:

Column name	Type	Description
prediction_score	Double	The predicted scores column. In a linear model, the feature values and model weight values are summed up or multiplied to obtain the predicted scores. In a scorecard model, if score transformation is performed, the transformed scores are input into this column.
prediction_prob	Double	The probability values of positive samples in binary classification. The probability values are transformed from the original scores (before score transformation) by using the sigmoid function.
prediction_detail	String	The probability values of positive and negative samples described in JSON strings. Value 0 represents negative and value 1 represents positive. Example: { "0" :0.1813110520," 1" :0.8186889480}.

PAI command

```
pai -name=lm_predict
  -project=algo_public
  -DinputFeatureTableName=input_data_table
  -DinputModelTableName=input_model_table
  -DmetaColNames=sample_key,label
  -DfeatureColNames=fea1,fea2
  -DoutputTableName=output_score_table
```

Algorithm parameters

Parameter	Description	Valid value	Default
inputFeatureTableName	Required. The name of the input table that stores feature data.	N/A	N/A
inputFeatureTablePartitions	Optional. The partitions selected from the input table.	N/A	The whole table is selected.
inputModelTableName	Required. The name of the model table.	N/A	N/A

Parameter	Description	Valid value	Default
featureColNames	Optional. The names of the feature columns selected from the input table.	N/A	All columns are selected.
metaColNames	Optional. The names of the columns to be reserved in the result table.	N/A	The meta column is excluded. You can specify columns such as label and sample_id.
outputFeatureScore	Optional. It specifies whether to output variable scores to the result table.	true and false	false
outputTableName	Required. The name of the result table.	N/A	N/A
lifecycle	Optional. The lifecycle of the result table.	N/A	No lifecycle is set.
coreNum	Optional. The number of cores.	N/A	Automatically calculated.
memSizePerCore	Optional. The memory size per core.	N/A	Automatically calculated.

4.4.12.5. PSI

Population stability index (PSI) is an important metric to identify a shift in two samples of a population. For example, you can use it to measure whether the changes in the population within two months are stable. A PSI value smaller than 0.1 indicates insignificant changes. A PSI value between 0.1 and 0.25 indicates minor changes. A PSI value greater than 0.25 indicates major changes in the population.

When the changes in a population over time are unstable, you can use charts to identify the changes. You can use binning to discretize variables into multiple bins, calculate the number and proportion of the samples in each bin, and then display the statistics in a chart, as shown in the following figure.

This method can directly show whether a variable in two samples changes significantly. However, the shift in these changes cannot be measured by using this method. This means that the population stability cannot be automatically monitored. To resolve this issue, you can use the PSI component. Before you use the PSI component, you must use the Binning component to discretize the data. The formula for calculating PSI values is as follows:

Examples

The following figure shows a use case of the PSI component. The PSI component is connected to the Binning component and two sample datasets. You only need to specify the columns for PSI calculation in the PSI component.

Result

PAI command

```
PAI -name psi
-project algo_public
-DinputBaseTableName=psi_base_table
-DinputTestTableName=psi_test_table
-DoutputTableName=psi_bin_table
-DinputBinTableName=pai_index_table
-DfeatureColNames=fea1,fea2,fea3
-Dlifecycle=7
```

Algorithm parameters

Parameter	Description	Valid value	Default
inputBaseTableName	Required. The name of the base table. The shift of the population is calculated based on the samples in the base and test tables.	N/A	N/A
inputBaseTablePartitions	Optional. The partitions in the base table.	N/A	The whole table is selected.
inputTestTableName	Required. The name of the test table. The shift of the population is calculated based on the samples in the base and test tables.	N/A	N/A
inputTestTablePartitions	Optional. The partitions in the test table.	N/A	The whole table is selected.
inputBinTableName	Required. The name of the binning result table.	N/A	N/A
featureColNames	Optional. The features specified for PSI calculation.	N/A	All features are selected.
outputTableName	Required. The name of the output table (PSI statistics).	N/A	N/A
lifecycle	Optional. The lifecycle of the output table.	N/A	No lifecycle is set.
coreNum	Optional. The number of cores.	N/A	Automatically calculated.
memSizePerCore	Optional. The memory size per core.	N/A	Automatically calculated.

4.4.13. Video intelligence platform (must be separately activated)

4.4.13.1. Video preprocessing

4.4.13.1.1. Convert datasets to TFRecord files

EasyVision of Machine Learning Platform for AI allows you to convert datasets that are labeled by the data labeling module to TFRecord files. TFRecord files can be used to train models. You can use the TFRecord conversion component of EasyVision to convert labeled datasets that are stored in Object Storage Service (OSS) to TFRecord files and store the TFRecord files in OSS. If the datasets are labeled by using a method other than the data labeling module, you must first convert the labeled datasets to a format that is required by Machine Learning Platform for AI.

Machine Learning Platform for AI command example

Convert a labeled dataset for classification

```
pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=acs:ram::xxx:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dcmd convert
  -Dlabel_file 'oss://pai-vision-data-hz/data/pai_test/watchdog.csv'
  -Dconvert_param_config '
--class_list_file oss://pai-vision-data-hz/data/pai_test/watchdog_class_list.txt
--max_image_size 600
--write_parallel_num 8
--num_samples_per_tfrecord 128
--test_ratio 0.1
--model_type CLASSIFICATION
'
  -Doutput_tfrecord 'oss://pai-vision-data-hz/test/convert/watchdog_tfrecord/watchdog'
  -Dcluster='{
  \ "worker" : {
  \ "count" : 3,
  \ "cpu" : 800
  }
}'
```

Convert a labeled dataset for text detection or recognition

```

pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=acs:ram::xxx:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dcmd convert
  -Dlabel_file 'oss://pai-vision-data-hz/data/receipt_text/qince_data/qince_receipt_test.csv'
  -Dconvert_param_config '
    --model_type TEXT_END2END
    --default_class text
    --max_image_size 2000
    --char_replace_map_path oss://pai-vision-data-hz/data/receipt_text/qince_data/char_replace_map.csv
    --default_char_dict_path oss://pai-vision-data-hz/data/receipt_text/qince_data/char_dict
    --test_ratio 0.1
    --write_parallel_num 8
    --num_samples_per_tfrecored 64
    '
  -Doutput_tfrecored 'oss://pai-vision-data-hz/test/convert/receipt_text_end2end/'

```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
cmd	Yes	Set the value to convert.	String	convert
buckets	No	The OSS bucket where the input data is stored. You can specify multiple OSS buckets by separating them with commas (,). Each bucket name must end with a forward slash (/).	"oss://bucket_name/?role_arn=xxx&host=yyy" "oss://bucket_1/?role_arn=xxx&host=yyy,oss://bucket_2/"	""
label_file	Yes	The OSS path of the labeled CSV file.	oss://your_bucket/xxx.csv	Required
convert_param_config	No	The information about the conversion task. For more information, see the following table. You can also replace the convert_param_config parameter with the convert_config parameter.	--parama valuea --paramb valueb	""
convert_config	No	The OSS path of the conversion task profile.	String	""
output_tfrecored	No	The OSS path of the TFRecord file.	oss://your_dir/prefix	""

Parameter	Required	Description	Value format or sample value	Default value
cluster	No	The information about the workers that are used to perform conversion in a distributed manner. For more information, click here .	JSON string	<pre>{"worker": {"count": 3, "cpu": 800, "gpu": 0, "memory": 20000}}</pre>

convert_param_config

Parameter	Required	Description	Value format or sample value	Default value
-----------	----------	-------------	------------------------------	---------------

Parameter	Required	Description	Value format or sample value	Default value
model_type	Yes	<p>The type of model to which the converted data applies. Valid values:</p> <ul style="list-style-type: none"> • CLASSIFICATION: single-label or multi-label image classification • DETECTION: object detection • POLYGON_SEGMENTATION: semantic segmentation based on polygon labeling • SEGMENTATION: semantic segmentation • INSTANCE_SEGMENTATION: instance segmentation • TEXT_END2END: end-to-end optical character recognition (OCR) • TEXT_RECOGNITION: single-line text recognition • TEXT_DETECTION: text detection • VIDEO_CLASSIFICATION: video classification • SELF_DEFINED: custom conversion <p>If the value of the model_type parameter is set to TEXT_END2END or TEXT_RECOGNITION, the char_replace_map_path and default_char_dict_path parameters take effect.</p> <p>If the value of the model_type parameter is set to VIDEO_CLASSIFICATION, the decode_type, sample_fps, reshape_size, decode_batch_size, and decode_keep_size parameters take effect.</p>	String	N/A

Parameter	Required	Description	Value format or sample value	Default value
class_list_file	No	<p>The OSS path of the category file. The file contains a list of category names. The category name in each line may be presented in one of the following formats:</p> <ul style="list-style-type: none"> Category name Category name:Name of the mapping category <p>If the default_class parameter is set, you must set the class_list_file or class_list parameter.</p>	oss://path/to/your/classlit	""
class_list	No	<p>The list of category names. Separate category names with commas (,). If the default_class parameter is set, you must set the class_list_file or class_list parameter.</p>	String list	""
test_ratio	No	<p>The ratio that is used to divide the set of test data into different subsets. Valid values: 0 to 1. If the value is set to 0, the total set of test data is used for training. If the value is set to 0.1, 10% of the test data is used for verification.</p>	0.1	0.1
max_image_size	No	<p>The maximum pixel value for the longer side of the images. If you have set this parameter and the size of an image exceeds the upper limit, the image is resized and saved to the TFRecord file. This reduces storage space and accelerates data reading.</p>	INT	N/A
max_test_image_size	No	<p>The maximum pixel value for the longer side of the image that is used for testing. The value of this parameter is the same as that of the max_image_size parameter. This parameter is used to configure test data.</p>	INT	\${max_image_size}

Parameter	Required	Description	Value format or sample value	Default value
default_class	No	The name of the default category. Category names that are not included in the class_list parameter will be specified as the default category.	STRING	""
error_class	No	The name of the invalid category. Objects or bounding boxes that belong to this category are not used for training.	STRING	""
ignore_class	No	The name of the category that is used only to detect models. Bounding boxes that belong to this category are not used for training.	STRING	""
converter_class	No	The name of the conversion class. Valid values: <ul style="list-style-type: none"> • PaiConverter: converts the datasets to the Machine Learning Platform for AI format • QinceConverter: converts the datasets to the traditional labeling format • A custom class name 	STRING	PaiConverter
separator	No	The separator that is used in the split() method to break the label file into substrings.	STRING	""
image_format	No	The encoding format of the images in the TFRecord file.	STRING	jpg
read_parallel_num	No	The number of concurrent reads.	INT	10
write_parallel_num	No	The number of concurrent writes to the TFRecord file.	INT	1
num_samples_per_tfrecord	No	The number of images that are saved in each TFRecord file.	INT	256
user_defined_converter_path	No	The HTTP or OSS path of the user-defined converter code. Example: http://path/to/your/converter.py.	STRING	""

Parameter	Required	Description	Value format or sample value	Default value
user_defined_generator_path	No	The HTTP or OSS path of the user-defined generator code. Example: <code>http://path/to/your/generator.py</code> .	STRING	""
generator_class	No	The name of the user-defined generator class.	STRING	""
char_replace_map_path	No	The OSS path of the CSV file for character map replacement. The CSV file contains two columns named original and replaced. The strings in the original column are replaced by the strings in the replaced column.	STRING	""
default_char_dict_path	No	The OSS path of the file that is used to map characters to IDs. In the file, each character occupies a line. The ID of a character equals the line number minus 1.	STRING	""
decode_type	No	The method that is used to decode the videos. Valid values: <ul style="list-style-type: none"> • 1: intra only • 2: keyframe only • 3: without bidir • 4: decode all 	INT	4
sample_fps	No	The number of frames that are extracted for sampling per second.	FLOAT	5
reshape_size	No	The size of the output frames, in pixels.	INT	224
decode_batch_size	No	The number of frames that are decoded at a time.	INT	0
task_id	No	The ID of the Machine Learning Platform for AI labeling job, in the format of label-xxxx. If the manifest file contains a labeling job, this parameter is not required.	STRING	""

Custom conversion

Machine Learning Platform for AI command

```

pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=acs:ram::xxx:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dcmd convert
  -Dlabel_file 'oss://pai-vision-data-hz/data/image_mat/qince/imagenat.txt'
  -Dconvert_param_config '
    --model_type SELF_DEFINED
    --converter_class ImageMatConverter
    --user_defined_converter_path oss://pai-vision-data-hz/samples/code/imagenat_converter.py
    --user_defined_generator_path oss://pai-vision-data-hz/samples/code/custom_generator_example.py
    --generator_class CustomGenerator
    --max_image_size 1200
    --write_parallel_num 2
    --num_samples_per_tfreord 6
    --test_ratio 0.0
  '

```

To customize dataset conversion to TFRecord files, you must implement the following classes:

- A row class that inherits the DataRow class. The row class is used to parse data in each line of the label file and obtain image URLs and label information.
- A generator class that inherits the TFRecordGenerator class. The generator class is used to convert the data that is parsed by the row class to the tf.train.Example protocol buffers and specify the data format in TFRecord.
- A converter class that inherits the MultiProcConverter class. The converter class is used to load label files and configure train-test split policies to complete the conversion.

Example on custom_converter

```

from __future__ import unicode_literals
from tensorflow.python.platform import gfile
import tensorflow as tf
import numpy as np
import cv2
import json
import os
from easy_vision.python.protos.data_config_pb2 import DataConfig
from easy_vision.python.dataset_tools.tf_record_converter import MultiProcConverter
from easy_vision.python.dataset_tools.tf_record_converter import DataRow
from easy_vision.python.dataset_tools.converter_register import register_converter
import logging
import sys
try:
    reload(sys)
    sys.setdefaultencoding('utf8')
except:
    pass
class ImageMatRow(DataRow):
    """
    parse an image_path mask_path line to get image url and answer
    The following methods of DataRow are reimplemented:
    get_mask
    """
    def __init__(self, row):
        """
        Args:
            row: generate by csv.reader, array of columns in csv file
        """

```

```
    super(ImageMatRow, self).__init__(row)
def get_img_url(self):
    """ get img url from raw_data
    Return:
        url, string, path of the image
    """
    return self._row[0]
def get_anws(self):
    """ get anws string from anws field
    Return:
        anws, json object, depends on task
    """
    return self._row[1]
def get_mask_url(self):
    """ must be implemented for segmentation task
    Return:
        string, mask url
    """
    return self._row[1]
@register_converter()
class ImageMatConverter(MultiProcConverter):
    """ implement methods for converting a text line file into tfrecord
        required by image segmentation models.
    """
def __init__(self, data_config):
    """
    Args:
        data_config: DataConfig instance
    """
    super(ImageMatConverter, self).__init__(data_config)
def _check_lines(self, file_path):
    """ only check the first 1k lines, to save time.
    Args:
        file_path: string, input file path to be checked
    Raises:
        AssertionError, if separator['\t'] is not appear in one of the text lines
    """
    line_num = 1000
    with gfile.GFile(file_path, mode='r') as fin:
        for line_str in fin:
            line_str = line_str.strip()
            assert '\t' in line_str, 'separator[\t] is not in line: %s' % line_str
def split_train_test(self, input_path, output_dir, test_ratio=0.1):
    data_arr = []
    with gfile.GFile(input_path, mode='r') as fin:
        for line in fin:
            data_arr.append(line)
        np.random.shuffle(data_arr)
    train_num = int(len(data_arr) * (1-test_ratio))
    # save train input
    _, input_file = os.path.split(input_path)
    input_name, _ = os.path.splitext(input_file)
    train_input_path = output_dir.rstrip('/') + '/' + input_name + '_train.txt'
    with gfile.GFile(train_input_path, mode='w') as train_out:
        for line_str in data_arr[:train_num]:
            train_out.write(line_str)
    # save test input
    test_input_path = output_dir.rstrip('/') + '/' + input_name + '_test.txt'
```

```
test_input_path = output_dir.rstrip('/') + '/' + input_name + '_test.txt'
with gfile.GFile(test_input_path, mode='w') as test_out:
    for line_str in data_arr[train_num:]:
        test_out.write(line_str)
return train_input_path, test_input_path
def _load_input(self, input_path):
    """ load input from easy_vision.python.input_path(a text format file)
        each line is image_path + '\t' + label_path
        label_path store an image with alpha channel,
        the alpha channel store the class of the image
    Args:
        input_path: string, input file path
    """
    with gfile.GFile(input_path, mode='r') as f:
        for line in f:
            line = line.strip()
            line_tok = [x for x in line.split('\t') if x != '']
            yield ImageMatRow(line_tok)
def _save_for_fix(self, file_name):
    # currently no errors are checked and saved
    pass
```

Example on custom_generator

```
from __future__ import print_function
from __future__ import absolute_import
from __future__ import division
from __future__ import unicode_literals
import cv2
import tensorflow as tf
import numpy as np
import logging
import os
from easy_vision.python.utils import dataset_util
from easy_vision.python.dataset_tools import dataset_info as dataset_info_lib
from easy_vision.python.dataset_tools.tf_record_generator import TFRecordGenerator
class CustomGenerator(TFRecordGenerator):
    def _to_tf_example(self, img, row, url):
        mask_url = row.get_mask_url()
        mask_data = row.get_url_data(mask_url)
        if mask_data is None:
            logging.error('prefetch mask %s failed' % mask_url)
            return []
        mask = cv2.imdecode(np.fromstring(mask_data, dtype=np.uint8), cv2.IMREAD_UNCHANGED)
        if mask is None:
            logging.error('decode mask %s failed' % mask_url)
            return []
        mask = np.array(mask > 0, dtype=np.uint8)
        height, width = img.shape[:2]
        succeed, encoded = cv2.imencode('.' + self.image_format, img)
        succeed, mask_data = cv2.imencode('.png', mask)
        _, img_name = os.path.split(url)
        example_info = dataset_info_lib.DatasetInfo()
        example_info.add_single(
            {
                'image/height': height,
                'image/width': width,
            }
        )
        example = tf.train.Example(
            features=tf.train.Features(
                feature={
                    'image/encoded': dataset_util.bytes_feature(encoded.tobytes()),
                    'image/filename':
                        dataset_util.bytes_feature(img_name),
                    'image/dataset_name':
                        dataset_util.bytes_feature(self._dataset_name.encode('utf-8')),
                    'image/height': dataset_util.int64_feature(height),
                    'image/width': dataset_util.int64_feature(width),
                    'image/source_id':
                        dataset_util.bytes_feature(url),
                    'image/format':
                        dataset_util.bytes_feature(self.image_format),
                    'image/segmentation/class/encoded': (
                        dataset_util.bytes_feature(mask_data.tobytes())),
                    'image/segmentation/class/format': dataset_util.bytes_feature('png')
                })
        ))
        return [(example, example_info)]
```

4.4.13.1.2. General video preprocessing

This topic provides an example on how to run a Machine Learning Platform for AI command to train a model to preprocess videos. This topic also describes the command parameters.

You can run the `ev_video_process` command to train a model to preprocess videos. When you run this command, you need only to set the common parameters without the need to know the rules or logic of the EasyVision profile.

Machine Learning Platform for AI command example

```

pai -name ev_video_process
  -Dinput_oss_file='oss://pai-vision-data-hz/data/pai_test/video_decode_test.list'
  -Doutput_oss_file='oss://pai-vision-data-hz/test/tmp/ev_video_decode_result.txt'
  -Doutput_dir='oss://pai-vision-data-hz/data/ucf101_train/frames/'
  -Ddecode_type=4
  -Dsample_fps=5
  -Dreshape_size=224
  -Ddecode_batch_size=10
  -Ddecode_keep_size=0
  -Dqueue_size=256
  -Dnum_worker=4
  -DcpuRequired=400
  -DgpuRequired=100
  -Dbuckets='YOUR_OSS_BUCKET_CONFIG'

```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
buckets	Yes	The endpoint of the Object Storage Service (OSS) bucket where your model is stored. This parameter is required when you use your own model to perform prediction.	"oss://bucket_name/?role_arn=xxx&host=yyy"	N/A
input_oss_file	Yes	The path of the input OSS file. Each line of the OSS object includes a video path.	oss://your_bucket/filelist.txt	N/A
output_oss_file	Yes	The path of the output OSS file where the preprocessing result is stored. The system may generate multiple result files and merge these result files into an output OSS file. The result files are prefixed with the name of the output OSS file. The number of result files is the same as the number of workers, which is specified by the num_worker parameter.	oss://your_bucket/result.txt	N/A

Parameter	Required	Description	Value format or sample value	Default value
output_dir	Yes	The directory that is used to store the frames extracted from the videos.	oss://your_bucket/your_dir	N/A
decode_type	No	The method that is used to decode the videos. Valid values: <ul style="list-style-type: none"> • 1: intra only • 2: keyframe only • 3: without bidir • 4: decode all 	INT	4
sample_fps	No	The number of frames that are extracted for sampling per second.	FLOAT	5
reshape_size	No	The size of the output frames, in pixels. If you set this parameter to -1, the frames are not resized.	INT	224
decode_batch_size	No	The number of frames that are decoded at a time.	INT	10
decode_keep_size	No	The number of overlapped frames in different batches.	INT	0
queue_size	No	The length of the preprocessing queue.	INT	256
thread_num	No	The number of threads on each worker.	INT	4
num_worker	No	The number of prediction workers.	INT	2
cpuRequired	No	The number of CPUs for a worker.	A value of 100 indicates one CPU.	800
gpuRequired	No	The number of GPUs for a worker.	A value of 100 indicates one GPU.	100

4.4.13.2. Offline training

4.4.13.2.1. Train a model for semantic image segmentation

This topic provides an example on how to run a Machine Learning Platform for AI command to train a model for semantic image segmentation in EasyVision. This topic also describes the command parameters.

A semantic segmentation model based on DeepLabv3 is implemented. For more information, see [Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation](#).

Machine Learning Platform for AI command example

The following code provides an example on the training of semantic image segmentation with a single GPU:

```

pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -DgpuRequired=100
  -Dcmd train
  -Dparam_config '
    --model_type DeeplabV3
    --backbone resnet_v1_50
    --backbone_feature_stride 16
    --bn_trainable true
    --num_classes 21
    --num_epochs 1
    --model_dir oss://pai-vision-data-hz/test/deeplab_stage1
    --train_data oss://pai-vision-data-hz/data/test/pascal_voc_seg_aug/voc_ev_train.tfrecord
    --test_data oss://pai-vision-data-hz/data/test/pascal_voc_seg_aug/voc_ev_train.tfrecord
    --num_test_example 2
    --train_batch_size 6
    --test_batch_size 1
    --image_crop_size 513
    --lr_type polynomial_decay
    --initial_learning_rate 0.007
    --power 0.9
  '

```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
buckets	Yes	The endpoint of the Object Storage Service (OSS) bucket.	oss://pai-vision-data-hz/?role_arn=acs:ram::1217060697xxxx:role/pai-vision-hz&host=cn-hangzhou.oss.aliyun-inc.com	N/A
cluster	No	The configuration of parameters that are used for distributed training.	JSON string	""
gpuRequired	No	Specifies whether to use GPUs. By default, each worker uses one GPU. If you set this parameter to 200, each worker uses two GPUs.	100	100
cmd	Yes	The type of the EasyVision task. To train a model, you must set this parameter to train.	train	N/A

Parameter	Required	Description	Value format or sample value	Default value
param_config	Yes	The configuration of parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. For more information, see the following table.	STRING	N/A

param_config

The param_config parameter contains the parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. The following code provides an example on the param_config parameter:

```
-Dparam_config = '  
--backbone resnet_v1_50  
--num_classes 200  
--model_dir oss://your/bucket/exp_dir  
'
```

Note The values of all the string parameters in the param_config parameter are not enclosed in double quotation marks (") or single quotation marks (').

The following table describes all parameters that are contained in the param_config parameter.

Parameter	Required	Description	Value format or sample value	Default value
model_type	Yes	The type of the model to train. Set this parameter to DeeplabV3.	STRING	N/A
backbone	Yes	The name of the backbone network that is used by the model. Valid values: <ul style="list-style-type: none"> resnet_v1_50 resnet_v1_101 resnet_v1a_18 resnet_v1a_34 resnet_v1d_50 resnet_v1d_101 xception_41 xception_65 xception_71 	STRING	N/A
weight_decay	No	The value of L2 regularization.	FLOAT	1e-4
num_classes	Yes	The number of categories, including background categories.	21	N/A

Parameter	Required	Description	Value format or sample value	Default value
backbone_feature_stride	No	The feature downsampling step size of the backbone network.	INT	16
bn_trainable	No	Specify whether the batch normalization (BN) layer is trainable. If the value of the train_batch_size parameter is greater than 8, set the bn_trainable parameter to true.	BOOL	true
image_crop_size	No	The size of the image after cropping.	INT	513
optimizer	No	The type of the optimizer. Valid values: <ul style="list-style-type: none"> momentum: stochastic gradient descent (SGD) with momentum adam 	STRING	momentum
lr_type	No	The policy that is used to adjust the learning rate. Valid values: <ul style="list-style-type: none"> exponential_decay: the exponential decay. polynomial_decay: the polynomial decay. <p>If you set the lr_type parameter to polynomial_decay, the num_steps parameter is automatically set to the total number of training iterations. The value of the end_learning_rate parameter is automatically set to one thousandth of the value of the initial_learning_rate parameter.</p> <ul style="list-style-type: none"> manual_step: The learning rate of each epoch is manually adjusted. <p>If you set the lr_type parameter to manual_step, you must set the decay_epochs parameter to specify the epochs for which you want to adjust the learning rate. You must also set the learning_rates parameter to specify the learning rate as needed.</p> <ul style="list-style-type: none"> cosine_decay: The learning rate of each epoch is adjusted based on the cosine curve until the learning rate decreases to 0. For more information, see SGDR: Stochastic Gradient Descent with Warm Restarts. 	STRING	exponential_decay

Parameter	Required	Description	Value format or sample value	Default value
initial_learning_rate	No	The initial learning rate.	FLOAT	0.01
decay_epochs	No	If you set the lr_type parameter to exponential_decay, the decay_epochs parameter is equivalent to the decay_steps parameter of tf.train.exponential.decay. In this case, the decay_epochs parameter specifies the epoch interval at which you want to adjust the learning rate. The system automatically converts the value of the decay_epochs parameter to the value of the decay_steps parameter based on the total number of training data entries. Typically, you can set the decay_epochs parameter to half of the total number of epochs. For example, if the total number of epochs is 20, you can set this parameter to 10. If you set the lr_type parameter to manual_step, the decay_epochs parameter specifies the epochs for which you want to adjust the learning rate. For example, a value of 16 18 indicates that you want to adjust the learning rate for the 16th and 18th epochs. Typically, if the total number of epochs is N, you can set the two values of the decay_epochs parameter to $8/10 \times N$ and $9/10 \times N$.	INTEGER list. Sample value: 20 20 40 60	20
decay_factor	No	The decay rate. This parameter is equivalent to the decay_factor parameter of tf.train.exponential.decay.	FLOAT	0.95
staircase	No	Specifies whether the learning rate changes based on the decay_epochs parameter. This parameter is equivalent to the staircase parameter of tf.train.exponential.decay.	BOOL	true
power	No	The power of the polynomial. This parameter is equivalent to the power parameter of tf.train.polynomial.decay.	FLOAT	0.9

Parameter	Required	Description	Value format or sample value	Default value
learning_rates	No	The learning rate that you want to set for the specified epochs. If you set the lr_type parameter to manual_step, this parameter is required. If you want to adjust the learning rate for two epochs, specify two learning rates in the value. For example, if the decay_epochs parameter is set to 20 40, you must specify two learning rates in the value of the learning_rates parameter, such as 0.001 0.0001. This indicates that the learning rate of the 20th epoch is adjusted to 0.001 and the learning rate of the 40th epoch is adjusted to 0.0001. We recommend that you adjust the learning rate to one tenth, one hundredth, and one thousandth of the initial learning rate in sequence.	FLOAT list	N/A
lr_warmup	No	Specifies whether to warm up the learning rate.	BOOL	false
lr_warm_up_epochs	No	The number of epochs for which you want to warm up the learning rate.	FLOAT	1
train_data	Yes	The OSS path of the data that is used to train the model.	oss://path/to/train_*.tfrecord	N/A
test_data	Yes	The OSS path of the data that is evaluated during the training.	oss://path/to/test_*.tfrecord	N/A
train_batch_size	Yes	The size of the data that is used to train the model in the current batch.	INT. Sample value: 32	N/A
test_batch_size	Yes	The size of the data that is evaluated in the current batch.	INT. Sample value: 32	N/A
train_num_readers	No	The number of concurrent threads that are used to read the training data.	INT	4
model_dir	Yes	The OSS path of the model.	oss://path/to/model	N/A
pretrained_model	No	The OSS path of the pretrained model. If this parameter is specified, the actual model is finetuned based on the pretrained model.	oss://pai-vision-data-hz/pretrained_models/inception_v4.ckpt	""
use_pretrained_model	No	Specifies whether to use a pretrained model.	BOOL	true

Parameter	Required	Description	Value format or sample value	Default value
num_epochs	Yes	The number of times the data is iterated for the training. A value of 1 indicates that all data is iterated once for the training.	INT. Sample value: 40	N/A
num_test_example	No	The number of data entries that are evaluated during the training. A value of -1 indicates that all training data is evaluated.	INT. Sample value: 2000	-1
num_visualizations	No	The number of data entries that can be visualized during the evaluation.	INT	10
save_checkpoint_epochs	No	The epoch interval at which a checkpoint is saved. A value of 1 indicates that a checkpoint is saved each time an epoch is complete.	INT	1
save_summary_epochs	No	The epoch interval at which a summary is saved. A value of 0.01 indicates that a summary is saved each time 1% of the training data is iterated.	FLOAT	0.01
num_train_images	No	The total number of data entries that are used for the training. If you use custom TFRecord files to train the model, this parameter is required.	INT	0
label_map_path	No	The category mapping file. If you use custom TFRecord files to train the model, this parameter is required.	STRING	""

4.4.13.2.2. Train an image detection model

This topic provides an example on how to run a Machine Learning Platform for AI command to train an image detection model. This topic also describes the command parameters.

EasyVision allows you to run a command to train an image detection model. You can use the `-Dparam_config` parameter to set common parameters without the need to know the rules or logic of the EasyVision profile. If you need advanced parameters to train an image detection model, you can use the `-Dconfig` parameter to pass the profile to EasyVision. The following table describes the models that are supported for image detection training.

Model	Backbone	Whether FPN is supported
-------	----------	--------------------------

Model	Backbone	Whether FPN is supported
Faster R-CNN	<ul style="list-style-type: none"> • resnet_v1_50 • resnet_v1_101 • resnet_v1a_18 • resnet_v1a_34 • resnet_v1d_50 • resnet_v1d_101 • resnet_v1d_152 	Yes
R-FCN	<ul style="list-style-type: none"> • resnet_v1_50 • resnet_v1_101 • resnet_v1a_18 • resnet_v1a_34 • resnet_v1d_50 • resnet_v1d_101 • resnet_v1d_152 	No
SSD	<ul style="list-style-type: none"> • resnet_v1_50 • resnet_v1d_50 	Yes
SSD	vgg16_reduce_fc	No
SSD	mobilenet_v1	No

Machine Learning Platform for AI command example

- The following code provides an example on how to train a single-shot detection (SSD) model by using a single GPU:

```

pai -name easy_vision
  -Dbuckets='oss://YOUR_BUCKET_NAME/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -DgpuRequired=100
  -Dcmd train
  -Dparam_config '
    --model_type SSD
    --backbone resnet_v1_50
    --num_classes 20
    --model_dir oss://YOUR_BUCKET_NAME/test/ssd_resnet50
    --train_data oss://pai-vision-data-hz/data/voc0712_tfrecord/voc0712_part_*.tfrecord
    --test_data oss://pai-vision-data-hz/data/voc0712_tfrecord/VOC2007_test.tfrecord
    --num_test_example 2
    --train_batch_size 32
    --test_batch_size 1
    --image_sizes 300
    --lr_type exponential_decay
    --initial_learning_rate 0.001
    --decay_epochs 20
    --staircase true
  '

```

- The following code provides an example on how to train a Faster R-CNN model or an R-FCN model by using

a single GPU:

```
pai -name easy_vision
  -Dbuckets='oss://YOUR_BUCKET_NAME/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -DgpuRequired=100
  -Dcmd train
  -Dparam_config '
    --model_type FasterRCNN
    --backbone resnet_v1_50
    --num_classes 20
    --model_dir oss://YOUR_BUCKET_NAME/test/ssd_fpn_resnet50
    --train_data oss://pai-vision-data-hz/data/voc0712_tfrecord/voc0712_part_*.tfrecord
    --test_data oss://pai-vision-data-hz/data/voc0712_tfrecord/VOC2007_test.tfrecord
    --num_test_example 2
    --train_batch_size 32
    --test_batch_size 1
    --image_min_sizes 600
    --image_max_sizes 1024
    --lr_type exponential_decay
    --initial_learning_rate 0.001
    --decay_epochs 20
    --staircase true
  '
```

- The following code provides an example on how to train an SSD model by using multiple GPUs:

```
pai -name easy_vision
  -Dbuckets='oss://YOUR_BUCKET_NAME/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dcmd train
  -Dcluster='{
    \"ps\": {
      \"count\": 1,
      \"cpu\": 600
    },
    \"worker\": {
      \"count\": 3,
      \"cpu\": 800,
      \"gpu\": 100
    }
  }'
  -Dparam_config '
    --model_type SSD
    --backbone resnet_v1_50
    --num_classes 20
    --model_dir oss://YOUR_BUCKET_NAME/test/ssd_fpn_resnet50
    --train_data oss://pai-vision-data-hz/data/voc0712_tfrecord/voc0712_part_*.tfrecord
    --test_data oss://pai-vision-data-hz/data/voc0712_tfrecord/VOC2007_test.tfrecord
    --num_test_example 2
    --train_batch_size 32
    --test_batch_size 1
    --image_sizes 300
    --lr_type exponential_decay
    --initial_learning_rate 0.001
    --decay_epochs 20
    --staircase true
  '
```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
buckets	Yes	The endpoint of the Object Storage Service (OSS) bucket.	oss://pai-vision-data-hz/?role_arn=acs:ram::12170606971xxx:role/pai-vision-hz&host=cn-hangzhou.oss.aliyun-inc.com	N/A
cluster	No	The configuration of parameters that are used for distributed training.	JSON string	""
gpuRequired	No	Specifies whether to use GPUs. By default, each worker uses one GPU. If you set this parameter to 200, each worker uses two GPUs.	100	100
cmd	Yes	The type of the EasyVision task. To train a model, you must set this parameter to train.	train	N/A
param_config	Yes	The configuration of parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. For more information, see the following table.	STRING	N/A

param_config

The param_config parameter contains the parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. The following code provides an example on the param_config parameter:

```
-Dparam_config='
--backbone resnet_v1_50
--num_classes 200
--model_dir oss://your/bucket/exp_dir
'
```

 **Note** Do not enclose the values of all the string parameters in the param_config parameter in double quotation marks (") or single quotation marks (').

Parameter	Required	Description	Value format or sample value	Default value
model_type	Yes	The type of the model to train. Valid values: <ul style="list-style-type: none"> SSD FasterRCNN RFCN 	STRING	N/A
backbone	Yes	The name of the backbone network that is used by the model. Valid values: <ul style="list-style-type: none"> mobilenet_v1 resnet_v1_50 resnet_v1_101 resnet_v1a_18 resnet_v1a_34 resnet_v1d_50 resnet_v1d_101 vgg16_reduce_fc 	STRING	N/A
weight_decay	No	The value of L2 regularization.	FLOAT	1e-4
use_fpn	No	Specifies whether to use Feature Pyramid Network (FPN).	BOOL	false
num_classes	No	The number of categories, excluding background categories.	21	N/A
anchor_scales	No	The size of the anchor box. The size of the anchor box is the same as that of the input image where the anchor box resides after the image is resized. If you use an SSD model, this parameter is not required. If you use FPN, set this parameter to the size of the anchor box in the layer that has the highest resolution. The total number of layers is five. The size of the anchor box in a layer is twice as that in the previous layer. For example, if the size of the anchor box in the first layer is 32, the sizes of the anchor boxes in the next four layers are 64, 128, 256, and 512. If you use the Faster R-CNN model or the R-FCN model without FPN support, you can specify multiple anchor sizes as needed. For example, you can set the anchor_scales parameter to 128 256 512.	FLOAT list Sample value: 32 (single scale) or 128 256 512 (multiple scales)	<ul style="list-style-type: none"> SSD: The default value is 0.1, 0.2, 0.37, 0.54, 0.71, 0.88, 0.96, or 1.0 times of the size of the input image. FPN: 32 Faster R-CNN with FPN support: 32 Faster R-CNN or R-FCN: [128 256 512]
anchor_ratios	No	The ratios of the width to the height of the anchor boxes.	FLOAT list	0.5 1 2

Parameter	Required	Description	Value format or sample value	Default value
image_sizes	No	The size of the images after they are resized. This parameter takes effect only when an SSD model is used. The value of this parameter is a list that contains two numbers, which indicate the height and width.	FLOAT list	300 300
image_min_sizes	No	The length of the shorter side of images after they are resized. This parameter is used for Faster R-CNN and R-FCN models. If you specify multiple lengths for the shorter sides of images in the value of this parameter, the last length but one is used to train the model, whereas the last one is used to evaluate the model. If you specify only one length for the shorter sides of images, this length is used for both training and evaluation.	FLOAT list	600
image_max_sizes	No	The length of the longer side of images after they are resized. This parameter is used for Faster R-CNN and R-FCN models. If you specify multiple lengths for the longer sides of images in the value of this parameter, the last length but one is used to train the model, whereas the last one is used to evaluate the model. If you specify only one length for the longer sides of images, this length is used for both training and evaluation.	FLOAT list	1024
optimizer	No	The type of the optimizer. Valid values: <ul style="list-style-type: none"> momentum: stochastic gradient descent (SGD) with momentum adam 	STRING	momentum

Parameter	Required	Description	Value format or sample value	Default value
lr_type	No	<p>The policy that is used to adjust the learning rate. Valid values:</p> <ul style="list-style-type: none"> exponential_decay: the exponential decay. polynomial_decay: the polynomial decay. <p>If you set the lr_type parameter to polynomial_decay, the num_steps parameter is automatically set to the total number of training iterations. The value of the end_learning_rate parameter is automatically set to one thousandth of the value of the initial_learning_rate parameter.</p> <ul style="list-style-type: none"> manual_step: The learning rate of each epoch is manually adjusted. <p>If you set the lr_type parameter to manual_step, you must set the decay_epochs parameter to specify the epochs for which you want to adjust the learning rate. You must also set the learning_rates parameter to specify the learning rate as needed.</p> <ul style="list-style-type: none"> cosine_decay <p>The learning rate of each epoch is adjusted based on the cosine curve until the learning rate decreases to 0. For more information, see SGDR: Stochastic Gradient Descent with Warm Restarts. If you set the lr_type parameter to cosine_decay, you must set the decay_epochs parameter to specify the epochs for which you want to adjust the learning rate.</p>	STRING	exponential_decay
initial_learning_rate	No	The initial learning rate.	Floating point	0.01

Parameter	Required	Description	Value format or sample value	Default value
decay_epochs	No	If you set the lr_type parameter to exponential_decay, the decay_epochs parameter is equivalent to the decay_steps parameter of tf.train.exponential.decay. In this case, the decay_epochs parameter specifies the epoch interval at which you want to adjust the learning rate. The system automatically converts the value of the decay_epochs parameter to the value of the decay_steps parameter based on the total number of training data entries. Typically, you can set the decay_epochs parameter to half of the total number of epochs. For example, if the total number of epochs is 20, you can set this parameter to 10. If you set the lr_type parameter to manual_step, the decay_epochs parameter specifies the epochs for which you want to adjust the learning rate. For example, a value of 16 18 indicates that you want to adjust the learning rate for the 16th and 18th epochs. Typically, if the total number of epochs is N, you can set the two values of the decay_epochs parameter to $8/10 \times N$ and $9/10 \times N$.	INTEGER list. Sample value: 20 20 40 60	20
decay_factor	No	The decay rate. This parameter is equivalent to the decay_factor parameter of tf.train.exponential.decay.	FLOAT	0.95
staircase	No	Specifies whether the learning rate changes based on the decay_epochs parameter. This parameter is equivalent to the staircase parameter of tf.train.exponential.decay.	BOOL	true
power	No	The power of the polynomial. This parameter is equivalent to the power parameter of tf.train.polynomial.decay.	FLOAT	0.9

Parameter	Required	Description	Value format or sample value	Default value
learning_rates	No	The learning rate that you want to set for the specified epochs. This parameter is required when you set the lr_type parameter to manual_step. If you want to adjust the learning rate for two epochs, specify two learning rates in the value. For example, if the decay_epochs parameter is set to 20 40, you must specify two learning rates in the learning_rates parameter, such as 0.001 0.0001. This indicates that the learning rate of the 20th epoch is adjusted to 0.001 and the learning rate of the 40th epoch is adjusted to 0.0001. We recommend that you adjust the learning rate to one tenth, one hundredth, and one thousandth of the initial learning rate in sequence.	FLOAT list	N/A
lr_warmup	No	Specifies whether to warm up the learning rate.	BOOL	false
lr_warm_up_epochs	No	The number of epochs for which you want to warm up the learning rate.	FLOAT	1
train_data	Yes	The OSS path of the data that is used to train the model.	oss://path/to/train_*.tfrecord	N/A
test_data	Yes	The OSS path of the data that is evaluated during the training.	oss://path/to/test_*.tfrecord	N/A
train_batch_size	Yes	The size of the data that is used to train the model in the current batch.	INT. Sample value: 32	N/A
test_batch_size	Yes	The size of the data that is evaluated in the current batch.	INT. Sample value: 32	N/A
train_num_readers	No	The number of concurrent threads that are used to read the training data.	INT	4
model_dir	Yes	The OSS path of the model.	oss://path/to/model	N/A
pretrained_model	No	The OSS path of the pretrained model. If this parameter is specified, the actual model is finetuned based on the pretrained model.	oss://pai-vision-data-sh/pretrained_models/inception_v4.ckpt	""
use_pretrained_model	No	Specifies whether to use a pretrained model.	BOOL	true

Parameter	Required	Description	Value format or sample value	Default value
num_epochs	Yes	The number of training iterations. A value of 1 indicates that all data is iterated once for the training.	INT. Sample value: 40	N/A
num_test_example	No	The number of data entries that are evaluated during the training. A value of -1 indicates that all training data is evaluated.	INT. Sample value: 2000	-1
num_visualizations	No	The number of data entries that can be visualized during the evaluation.	INT	10
save_checkpoint_epochs	No	The epoch interval at which a checkpoint is saved. A value of 1 indicates that a checkpoint is saved each time an epoch is complete.	INT	1
save_summary_epochs	No	The epoch interval at which a summary is saved. A value of 0.01 indicates that a summary is saved each time 1% of the training data is iterated.	FLOAT	0.01
num_train_images	No	The total number of data entries that are used for the training. If you use custom TFRecord files to train the model, this parameter is required.	INT	0
label_map_path	No	The category mapping file. If you use custom TFRecord files to train the model, this parameter is required.	STRING	""

4.4.13.2.3. Train an image classification model

This topic provides examples on how to run a Machine Learning Platform for AI command to train an image classification model in EasyVision. This topic also describes the command parameters.

EasyVision allows you to run a command to train an image classification model. When you run the command, you need only to set the common classification parameters without the need to know the rules or logic of the EasyVision profile.

Machine Learning Platform for AI command example

- Training on a single server

```
pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -DgpuRequired=100
  -Dcmd train
  -Dparam_config '
    --model_type Classification
    --backbone inception_v4
    --num_classes 10
    --num_epochs 1
    --model_dir oss://pai-vision-data-hz/test/cifar_inception_v4
    --use_pretrained_model true
    --train_data oss://pai-vision-data-hz/data/test/cifar10/*.tfrecord
    --test_data oss://pai-vision-data-hz/data/test/cifar10/*.tfrecord
    --num_test_example 20
    --train_batch_size 32
    --test_batch_size=32
    --image_size 299
    --initial_learning_rate 0.01
    --staircase true '
  '
```

- Training on multiple servers

```
pai -name easy_vision
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dcmd train
  -Dcluster='{
    \"ps\": {
      \"count\": 1,
      \"cpu\": 600
    },
    \"worker\": {
      \"count\": 3,
      \"cpu\": 800,
      \"gpu\": 100
    }
  }'
  -Dparam_config='
    --model_type Classification
    --backbone inception_v4
    --num_classes 10
    --num_epochs 1
    --model_dir oss://pai-vision-data-hz/test/cifar_inception_v4_dis
    --use_pretrained_model true
    --train_data oss://pai-vision-data-hz/data/test/cifar10/*.tfrecord
    --test_data oss://pai-vision-data-hz/data/test/cifar10/*.tfrecord
    --num_test_example 20
    --train_batch_size 32
    --test_batch_size=32
    --image_size 299
    --initial_learning_rate 0.01
    --staircase true
  '
```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
buckets	Yes	The endpoint of the Object Storage Service (OSS) bucket.	oss://pai-vision-data-hz/?role_arn=acs:ram::12170606971xxx:role/pai-vision-hz&host=cn-hangzhou.oss.aliyun-inc.com	N/A
cluster	No	The configuration of parameters that are used for distributed training.	JSON string	""
gpuRequired	No	Specifies whether to use GPUs. By default, each worker uses one GPU. If you set this parameter to 200, each worker uses two GPUs.	100	100
cmd	Yes	The type of the EasyVision task. To train a model, you must set this parameter to train.	train	N/A
param_config	Yes	The configuration of parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. For more information, see the following table.	STRING	N/A

param_config

The param_config parameter contains the parameters that are used for model training. The format of the param_config parameter is the same as that of the ArgumentParser() object in Python. The following code provides an example of the param_config parameter:

```
-Dparam_config='
--model_type Classification
--backbone inception_v4
--num_classes 200
--model_dir oss://your/bucket/exp_dir
'
```

 **Note** Do not enclose the values of all the string parameters in the param_config parameter in double quotation marks (") or single quotation marks (').

Parameter	Required	Description	Value format or sample value	Default value
model_type	Yes	The type of the model to train. To train a model for multi-label image classification, set this parameter to Classification.	STRING	N/A

Parameter	Required	Description	Value format or sample value	Default value
backbone	No	<p>The name of the backbone network that is used by the model. Valid values:</p> <ul style="list-style-type: none"> • vgg_16 • vgg_19 • inception_v1 • inception_v2 • inception_v3 • inception_v4 • mobilenet_v1 • mobilenet_v2 • mobilenet_v3 • resnet_v1_50 • resnet_v1_101 • resnet_v1_152 • resnet_v2_50 • resnet_v2_101 • resnet_v2_152 • resnet_v1a_18 • resnet_v1a_34 • resnet_v1a_50 • resnet_v1a_101 • resnet_v1a_152 • resnet_v1b_50 • resnet_v1b_101 • resnet_v1b_152 • resnet_v1c_50 • resnet_v1c_101 • resnet_v1c_152 • resnet_v1d_50 • resnet_v1d_101 • resnet_v1d_152 • efficientnet-b0 • efficientnet-b1 • efficientnet-b2 • efficientnet-b3 • efficientnet-b4 • efficientnet-b5 • efficientnet-b6 • efficientnet-b7 • efficientnet-b8 	STRING	inception_v4
weight_decay	No	The value of L2 regularization.	FLOAT	1e-4

Parameter	Required	Description	Value format or sample value	Default value
num_classes	Yes	The number of categories.	INT	N/A
hidden_size	No	The size of the custom Fiber Channel (FC) layer. A value of -1 indicates that the size of the original FC layer is used.	INT	-1
image_size	No	The size of the images after they are resized.	INT	224
use_crop	No	Specifies whether to crop images for data enhancement.	BOOL	true
crop_min_area	No	The minimum proportion of the original image area that is occupied by the crop box when you set the use_crop parameter to true.	FLOAT	0.7
eval_each_category	No	Specifies whether to separately evaluate each category.	BOOL	false
optimizer	No	The type of the optimizer. Valid values: <ul style="list-style-type: none"> momentum: stochastic gradient descent (SGD) with momentum adam 	STRING	momentum

Parameter	Required	Description	Value format or sample value	Default value
lr_type	No	<p>The policy that is used to adjust the learning rate. Valid values:</p> <ul style="list-style-type: none"> exponential_decay: the exponential decay. polynomial_decay: the polynomial decay. <p>If you set the lr_type parameter to polynomial_decay, the num_steps parameter is automatically set to the total number of training iterations. The value of the end_learning_rate parameter is automatically set to one thousandth of the value of the initial_learning_rate parameter.</p> <ul style="list-style-type: none"> manual_step: The learning rate of each epoch is manually adjusted. <p>If you set the lr_type parameter to manual_step, you must set the decay_epochs parameter to specify the epochs for which you want to adjust the learning rate. You must also set the learning_rates parameter to specify the learning rate as needed.</p> <ul style="list-style-type: none"> cosine_decay: The learning rate of each epoch is adjusted based on the cosine curve. For more information, see SGDR: Stochastic Gradient Descent with Warm Restarts. 	STRING	exponential_decay
initial_learning_rate	No	The initial learning rate.	FLOAT	0.01

Parameter	Required	Description	Value format or sample value	Default value
decay_epochs	No	If you set the lr_type parameter to exponential_decay, the decay_epochs parameter is equivalent to the decay_steps parameter of tf.train.exponential.decay. In this case, the decay_epochs parameter specifies the epoch interval at which you want to adjust the learning rate. The system automatically converts the value of the decay_epochs parameter to the value of the decay_steps parameter based on the total number of training data entries. Typically, you can set the decay_epochs parameter to half of the total number of epochs. For example, if the total number of epochs is 20, you can set this parameter to 10. If you set the lr_type parameter to manual_step, the decay_epochs parameter specifies the epochs for which you want to adjust the learning rate. For example, a value of 16 18 indicates that you want to adjust the learning rate for the 16th and 18th epochs. Typically, if the total number of epochs is N, you can set the two values of the decay_epochs parameter to $8/10 \times N$ and $9/10 \times N$.	INTEGER list. Sample value: 20 20 40 60	20
decay_factor	No	The decay rate. This parameter is equivalent to the decay_factor parameter of tf.train.exponential.decay.	FLOAT	0.95
staircase	No	Specifies whether the learning rate changes based on the decay_epochs parameter. This parameter is equivalent to the staircase parameter of tf.train.exponential.decay.	BOOL	true
power	No	The power of the polynomial. This parameter is equivalent to the power parameter of tf.train.polynomial.decay.	FLOAT	0.9

Parameter	Required	Description	Value format or sample value	Default value
learning_rates	No	The learning rate that you want to set for the specified epochs. If you set the lr_type parameter to manual_step, this parameter is required. If you want to adjust the learning rate for two epochs, specify two learning rates in the value. For example, if the decay_epochs parameter is set to 20 40, you must specify two learning rates in the value of the learning_rates parameter, such as 0.001 0.0001. This indicates that the learning rate of the 20th epoch is adjusted to 0.001 and the learning rate of the 40th epoch is adjusted to 0.0001. We recommend that you adjust the learning rate to one tenth, one hundredth, and one thousandth of the initial learning rate in sequence.	FLOAT list	N/A
lr_warmup	No	Specifies whether to warm up the learning rate.	BOOL	false
lr_warm_up_epochs	No	The number of epochs for which you want to warm up the learning rate.	FLOAT	1
train_data	Yes	The OSS path of the data that is used to train the model.	oss://path/to/train_*.tfrecord	N/A
test_data	Yes	The OSS path of the data that is evaluated during the training.	oss://path/to/test_*.tfrecord	N/A
train_batch_size	Yes	The size of the data that is used to train the model in the current batch.	INT. Sample value: 32	N/A
test_batch_size	Yes	The size of the data that is evaluated in the current batch.	INT. Sample value: 32	N/A
train_num_readers	No	The number of concurrent threads that are used to read the training data.	INT	4
model_dir	Yes	The OSS path of the model.	oss://path/to/model	N/A
pretrained_model	No	The OSS path of the pretrained model. If this parameter is specified, the actual model is finetuned based on the pretrained model.	oss://pai-vision-data-sh/pretrained_models/inception_v4.ckpt	""
use_pretrained_model	No	Specifies whether to use a pretrained model.	BOOL	true

Parameter	Required	Description	Value format or sample value	Default value
num_epochs	Yes	The number of times the data is iterated for the training. A value of 1 indicates that all data is iterated once for the training.	INT. Sample value: 40	N/A
num_test_example	No	The number of data entries that are evaluated during the training. A value of -1 indicates that all training data is evaluated.	INT. Sample value: 2000	-1
num_visualizations	No	The number of data entries that can be visualized during the evaluation.	INT	10
save_checkpoint_epochs	No	The epoch interval at which a checkpoint is saved. A value of 1 indicates that a checkpoint is saved each time an epoch is complete.	INT	1
save_summary_epochs	No	The epoch interval at which a summary is saved. A value of 0.01 indicates that a summary is saved each time 1% of the training data is iterated.	FLOAT	0.01
num_train_images	No	The total number of data entries that are used for the training. If you use custom TFRecord files to train the model, this parameter is required.	INT	0
label_map_path	No	The category mapping file. If you use custom TFRecord files to train the model, this parameter is required.	STRING	""

4.4.13.2.4. Train a video classification model

This topic provides an example on how to run a Machine Learning Platform for AI command to train a video classification model. This topic also describes the command parameters.

You can run the `ev_video_classification` command to train a model to classify videos. When you run this command, you need only to set the common classification parameters without the need to know the rules or logic of the EasyVision profile.

Machine Learning Platform for AI command example

- Training on a single server

```

pai -name ev_video_classification
  -Dbackbone='resnet_3d_50'
  -Dnum_classes=101
  -Dnum_epochs=2
  -Ddecay_epochs=1
  -Dsave_checkpoints_epochs=1
  -Dmodel_dir=oss://pai-vision-data-hz/test/UCF101_resnet_3d_50
  -Dpretrained_model='oss://pai-vision-data-hz/pretrained_models/resnet_3d_50/resnet_3d_50_model.cpk'
t'
  -Dtrain_data='oss://pai-vision-data-hz/data/ucf101/train/ucf101_train_*.tfrecord'
  -Dtest_data='oss://pai-vision-data-hz/data/ucf101/test/ucf101_test_0.tfrecord'
  -Dlabel_map_path='oss://pai-vision-data-hz/data/ucf101/test/ucf101_label_map.pbtxt'
  -Dnum_test_example=3783
  -Dtrain_batch_size=32
  -Dtest_batch_size=128
  -Dinitial_learning_rate=0.0001
  -Dstaircase=true
  -Dbuckets='YOUR_OSS_BUCKET_CONFIG'
  -DgpuRequired=100

```

- Training on multiple servers

```

pai -name ev_video_classification
  -Dbackbone='resnet_3d_50'
  -Dnum_classes=101
  -Dnum_epochs=2
  -Ddecay_epochs=1
  -Dsave_checkpoints_epochs=1
  -Dmodel_dir=oss://pai-vision-data-hz/test/UCF101_resnet_3d_50
  -Dtrain_data='oss://pai-vision-data-hz/data/ucf101/train/ucf101_train_*.tfrecord'
  -Dtest_data='oss://pai-vision-data-hz/data/ucf101/test/ucf101_test_0.tfrecord'
  -Dlabel_map_path='oss://pai-vision-data-hz/data/ucf101/test/ucf101_label_map.pbtxt'
  -Dpretrained_model='oss://pai-vision-data-hz/pretrained_models/resnet_3d_50/resnet_3d_50_model.cpk'
t'
  -Dnum_test_example=3783
  -Dtrain_batch_size=32
  -Dtest_batch_size=128
  -Dinitial_learning_rate=0.0001
  -Dstaircase=true
  -Dbuckets='YOUR_OSS_BUCKET_CONFIG'
  -Dcluster='{
    \"ps\": {
      \"count\": 1,
      \"cpu\": 600
    },
    \"worker\": {
      \"count\": 3,
      \"cpu\": 800,
      \"gpu\": 100
    }
  }'

```

Parameters

Parameter	Required	Description	Value format or sample value	Default value
backbone	Yes	The name of the backbone network that is used by the model. Valid values: <ul style="list-style-type: none"> c3d resnet_3d_18 resnet_3d_34 resnet_3d_50 resnet_3d_101 resnet_3d_152 resnet_3d_200 resnext_3d_50 resnext_3d_101 resnext_3d_152 resnext_3d_200 	STRING	resnet_3d_50
num_classes	Yes	The number of categories.	100	N/A
feature_name	No	The names of features. Example: ["resnet_3d_50/pool2","resnet_3d_50/scale5/conv5","resnet_3d_50/scale4/conv4","resnet_3d_50/scale3/conv3","resnet_3d_50/scale2/conv2","resnet_3d_50/pool1","resnet_3d_50/scale1/conv1"].	Character List	null
optimizer	No	The type of the optimizer. Valid values: <ul style="list-style-type: none"> momentum: stochastic gradient descent (SGD) with momentum adam 	STRING	momentum
decay_epochs	No	The epoch interval at which you want to adjust the learning rate. This parameter is equivalent to the decay_steps parameter of tf.train.exponential.decay.	INT	10
decay_factor	No	The decay rate. This parameter is equivalent to the decay_factor parameter of tf.train.exponential.decay.	FLOAT	0.95
staircase	No	Specifies whether the learning rate changes based on the decay_epochs parameter. This parameter is equivalent to the staircase parameter of tf.train.exponential.decay.	BOOL	true
cluster	No	The configuration of parameters that are used for distributed training.	JSON string	""

Parameter	Required	Description	Value format or sample value	Default value
gpuRequired	No	Specifies whether to use GPUs. By default, each worker uses one GPU. If you set this parameter to 200, each worker uses two GPUs.	INT	100
train_data	Yes	The Object Storage Service (OSS) path of the data that is used to train the model.	oss://path/to/train_*.tfrecord	N/A
test_data	Yes	The OSS path of the data that is evaluated during the training.	oss://path/to/test_*.tfrecord	N/A
train_batch_size	Yes	The size of the data that is used to train the model in the current batch.	INT. Sample value: 32	N/A
test_batch_size	Yes	The size of the data that is evaluated in the current batch.	INT. Sample value: 32	N/A
model_dir	Yes	The OSS path of the model.	oss://path/to/model	N/A
buckets	Yes	The endpoint of the OSS bucket.	oss://pai-vision-data-hz/?role_arn=acs:ram::12170606971xxx:role/pai-vision-hz&host=cn-hangzhou.oss.aliyun-inc.com	N/A
train_num_readers	No	The number of concurrent threads that are used to read the training data.	INT	4
pretrained_model	No	The OSS path of the pretrained model. If this parameter is specified, the actual model is finetuned based on the pretrained model.	oss://pai-vision-data-sh/pretrained_models/inception_v4.ckpt	""
initial_learning_rate	No	The initial learning rate.	FLOAT	0.0001
num_epochs	Yes	The number of training iterations. All data is iterated once for the training.	INT. Sample value: 20	N/A
num_test_example	Yes	The number of data entries that are evaluated during the training.	INT. Sample value: 2000	N/A
save_checkpoint_epochs	No	The epoch interval at which a checkpoint is saved.	INT	1

Parameter	Required	Description	Value format or sample value	Default value
label_map_path	Yes	The category mapping file.	STRING. Sample value: oss://path/to/*_labelmap.pbtxt	N/A

4.4.13.3. Offline prediction

4.4.13.3.1. General image prediction

The general image prediction component is used to make predictions. It reads data from MaxCompute tables or Object Storage Service (OSS) files for prediction, and writes the prediction results to the MaxCompute tables or OSS files. This topic provides examples on how to run a Machine Learning Platform for AI command to use the general image prediction component. This topic also describes how to define the input and output files and customize the process and prediction classes for different types of models.

Machine Learning Platform for AI command example

- Detect objects based on a region of interest

```
pai -name ev_predict
-Dmodel_path='oss://pai-vision-data-hz/pretrained_models/saved_models/faster_rcnn_with_roi_features_coco/'
-Dmodel_type='detector'
-Dinput_oss_file='oss://pai-vision-data-hz/data/pai_test/predict_oss_io_test.list'
-Doutput_oss_file='oss://pai-vision-data-hz/test/tmp/ev_predict_result.txt'
-Dimage_type='url'
-Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx:role/pai-vision-hz&host=cn-hangzhou.oss.aliyun-inc.com'
```

- Customize the process and predictor classes

```
pai -name ev_predict
-Dmodel_path='oss://pai-vision-data-hz/pretrained_models/pipeline_text_models/'
-Dmodel_type='self_define'
-Dinput_oss_file='oss://pai-vision-data-hz/data/pai_test/predict_oss_io_test.list'
-Doutput_oss_file='oss://pai-vision-data-hz/test/tmp/ev_predict_result.txt'
-Dimage_type='url'
-Duser_resource='oss://pai-vision-data-hz/data/pai_test/ev_predict/test_data/self_define/user_res.tar.gz'
-Duser_predictor_cls='user_predictor.MyPredictor'
-Duser_process_config='
[
{
  \"job_name\": \"myprocess\",
  \"num_threads\": 2,
  \"batch_size\": 1,
  \"class\": \"user_process.MyProcess\"
}
]'
-Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'
```

The code file that is used to customize the process class is *user_process.py*. The class name is *MyProcess*. The code file that is used to customize the predictor class is *user_predictor.py*. The class name is

MyPredictor. The *user_process.py* and *user_predictor.py* code files are compressed in the *user_res.tar.gz* package.

- Specify the input and output OSS files

```

pai -name ev_predict
-Dmodel_path='oss://pai-vision-data-hz/pretrained_models/pipeline_text_models/'
-Dmodel_type='self_define'
-Duser_resource='oss://your_predictor_res.tar.gz'
-Duser_predictor_cls='your_predictor.DummyModalPredictor'
-Dinput_oss_file='oss://pai-vision-data-hz/data/pai_test/predict_oss_io_test.list'
-Doutput_oss_file='oss://pai-vision-data-hz/test/tmp/ev_predict_multi_modal_result.json'
-Doutput_dir='oss://pai-vision-data-hz/test/tmp/multi_modal_output'
-Dimage_type='url'
-Dbuckets='oss://pai-vision-data-hz/?role_arn=xxx/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'

```

The code file that is used to customize the predictor class is *user_predictor.py*. The class name is *DummyModalPredictor*.

- Customize the process class

You can specify the input tables to read and process, and the output table to which the processed data is written. The following content provides an example on how to extract video features. In this example, two input tables are used. One of them contains the information about video frames. The other contains the IDs of the videos from which you want to extract features. You can use the following Machine Learning Platform for AI command to read data from the input tables and write the prediction results to the output table.

- Input table 1

Field	Type	Label	Comment
video_id	bigint		
frame_id	bigint		
image	string		
shot_id	bigint		
extra_info	string		
video_id_str	string		string type video id

- Input table 2

Field	Type	Label	Comment
video_id	bigint		

- Output table

Field	Type	Label	Comment
video_id	bigint		
frame_num	bigint		
feature	string		

- Machine Learning Platform for AI command

```
pai -name ev_predict
  -Dmodel_path='oss://pai-vision-data-hz/pretrained_models/saved_models/inception_v4_central_crop_1.0/'
  -Dmodel_type='feature_extractor'
  -Dfeature_name='PreLogitsFlatten'
  -Dinput_table='odps://sre_mpi_algo_dev/tables/zwm_dummy_table_data,odps://sre_mpi_algo_dev/tables/zwm_dummy_video_id'
  -Doutput_table='odps://sre_mpi_algo_dev/tables/zwm_dummy_result_video_feature'
  -Dimage_col='image'
  -Dimage_type='url'
  -Dreserved_columns='video_id'
  -Dresult_column='frame_num,feature'
  -Dbuckets='oss://pai-vision-data-hz/?role_arn=acs:ram::xxx:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'
  -Dauto_create_table=false
  -DuseSparseClusterSchema=true
  -DenableDynamicCluster=false
  -DautoEnablePsTaskFailover=false
  -DgpuRequired=50
  -Dnum_worker=1
  -Dqueue_size=12
  -Dbatch_size=1
  -Dpredict_thread_num=1
  -Dpreprocess_thread_num=3
  -Duser_resource http://pai-vision-data-hz.oss-cn-zhangjiakou.aliyuncs.com/data/pai_test/ev_predict/test_data/self_define/video_feature_extract_pad.py
  -Duser_build_process_fn 'video_feature_extract_pad.build_and_run_video_decode_process'
```

Parameters

Parameter	Required	Description	Value type	Default value
model_path	Yes	The OSS path of the model. Example: "oss://your_bucket/your_model_dir".	STRING	N/A

Parameter	Required	Description	Value type	Default value
model_type	Yes	<p>The type of the model. Valid values:</p> <ul style="list-style-type: none"> feature_extractor: feature extraction classifier: image classification detector: object detection text_detector: text detection text_recognizer: text line recognition text_spotter: end-to-end text recognition segmentor: semantic image segmentation self_define: custom prediction <p>If the model_type parameter is set to self_define, the predictor class that is specified by the user_predictor_cls parameter is loaded.</p>	STRING	N/A
buckets	No	<p>The information about the OSS buckets. If you use a custom model, you must specify the OSS bucket where your model is stored. Example: "oss://bucket_name/?role_arn=xxx&host=yyy".</p>	STRING	""
feature_name	No	<p>The name of the feature to extract. This parameter is required if the model_type parameter is set to feature_extractor. Example: resnet_v1_50/logits.</p>	STRING	""
input_table	No	<p>The name of the input table. For example, you can use a non-partitioned table "odps://prj_name/tables/table_name" or a partitioned table "odps://prj_name/tables/table_name/pt=xxx".</p>	STRING	""
image_col	No	<p>The name of the column that contains the image data.</p>	STRING	"image"

Parameter	Required	Description	Value type	Default value
image_type	No	The format of the image data. Valid values: <ul style="list-style-type: none"> base64: indicates that the table stores the image data in the Base64 format. url: indicates that the table stores the URLs or OSS paths of the images. 	STRING	"base64"
reserved_columns	No	The names of reserved data columns. Separate multiple names with commas (,). Example: "col1,col2,col3".	STRING	""
result_column	No	The name of the result column.	STRING	"prediction_result"
output_table	No	The output table. The value format is the same as that of the input_table parameter. If the specified output table does not exist, an output table is automatically created and partitioned. You can also create a partitioned table as the output table before you make the prediction.	STRING	""
lifecycle	No	The lifecycle of the output table.	INT	10
num_worker	No	The number of prediction workers. More workers can accelerate offline prediction.	INT	2
cpuRequired	No	The number of CPUs for a worker. A value of 100 indicates one CPU.	INT	1600
gpuRequired	No	The number of GPUs for each worker. A value of 100 indicates one GPU. You can use up to 100 GPUs for a worker. A value of 0 indicates that a CPU cluster is used.	INT	100
memory	No	The memory size of each worker, in MB.	INT	30000
input_oss_file	No	The path of the input OSS file. Each line in the input file can be in one of the following formats: <ul style="list-style-type: none"> The OSS path or URL of an image to predict. Example: oss://your_bucket/filelist.txt. A JSON string. 	STRING	""

Parameter	Required	Description	Value type	Default value
output_oss_file	No	The path of the output OSS file that is used to store the prediction results. The system may generate multiple result files and merge these result files into an output OSS file. The result files are prefixed with the name of the output OSS file. The number of result files is the same as the number of workers, which is specified by the num_worker parameter.	STRING	""
output_dir	No	The directory of the output file. If you customize the output format, all the result images are stored in this directory.	STRING	""
user_resource	No	The path to which you want to upload your resources. You can upload tar.gz, .zip, and .py files. OSS paths and HTTP URLs are supported. Example: oss://xxx/a.tar.gz or http://a.com/c.zip.	STRING	""
user_predictor_cls	No	The module path of the custom predictor class. For example, if you implement Predictor A in <i>module.py</i> , the module path of Predictor A is module.A.	STRING	""

Parameter	Required	Description	Value type	Default value
user_process_config	No	<p>The configurations of the custom process class. You can use the following fields to configure the process class. You can also configure other custom fields.</p> <ul style="list-style-type: none"> job_name: the name of the custom process class. num_threads: the number of concurrent threads that are used by the custom process class. batch_size: the batch size of the data to be processed. user_process_cls: the module path of the custom process class. For example, if you implement Process A in <i>module.py</i>, the module path of Process A is <i>module.A</i>. <p>Example: <pre>{["job_name":"myprocess","user_process_cls": module.ClassA"num_threads":2, "batch_size":1]}</pre> </p>	JSON string	""
queue_size	No	The length of the cache queue.	INT	1024
batch_size	No	The size of data that is used for prediction in the current batch.	INT	1
preprocess_thread_num	No	The number of concurrent threads that are used for preprocessing, such as image decoding and download.	INT	4
predict_thread_num	No	The number of concurrent threads that are used for prediction.	INT	2
is_input_video	No	Specifies whether the input files are video files.	BOOL	false
use_image_predictor	No	Specifies whether the predictor supports only image input.	BOOL	true
decode_type	No	<p>The method that is used to decode the videos. Valid values:</p> <ul style="list-style-type: none"> 1: intra only 2: keyframe only 3: without bidir 4: decode all 	INT	4

Parameter	Required	Description	Value type	Default value
sample_fps	No	The number of frames that are extracted for sampling per second.	FLOAT	5
reshape_size	No	The size of the output frames, in pixels. If you set this parameter to -1, the frames are not resized.	INT	-1
decode_batch_size	No	The number of frames that are decoded at a time.	INT	10
decode_keep_size	No	The number of overlapped frames in different batches.	INT	0
float_digits	No	The number of FLOAT digits for JSON serialization. For example, if you set this parameter to 3, the generated string contains three decimal places. A value of -1 indicates that the generated string uses the original precision.	INT	-1
mediaflow_class_name	No	The path of the Mediaflow class. Example: demo.mediaflow_caller.	STRING	""
self_defined_result_formatter	No	The class that is used to format the custom results. Example: a.table_result_format.	STRING	""
user_build_process_fn	No	The custom process function. Example: a.build_process_fn.	STRING	""
enableDynamicCluster	No	Specifies whether to enable the dynamic cluster feature. If this feature is enabled, the failover of a single worker is allowed. If task exceptions frequently occur, we recommend that you enable this feature.	BOOL	false
useSparseClusterSchema	No	If the enableDynamicCluster parameter is set to true, you must also set the useSparseClusterSchema parameter to true.	BOOL	false
enable_elastic_inference	No	Specifies whether to enable the elastic inference feature.	BOOL	false

Parameter	Required	Description	Value type	Default value
restore_works_dir	No	<p>The cache directory that is used to store the data being processed by each worker. When a worker failed, the data can be used to recover the worker. You can set this parameter to an OSS path or an Apsara File Storage for HDFS directory. If the enable_elastic_inference parameter is set to true, this parameter is required. Example: egoss://path/to/dir/hdfs://path/to/dir/.</p> <p>Note The value must end with a forward slash (/).</p>	STRING	""
slice_size	No	<p>The size of a data shard that a worker processes at a time. A large value may lead to out of memory (OOM) errors, whereas a small value may lead to low reading efficiency. We recommend that you set this parameter to 1024 for image processing, and 16 for video processing.</p>	INT	1024

Input data formats

- Input table

An input table can contain one or more columns. One of the columns must contain the image URLs or Base64-encoded binary image data. The data type of the column must be STRING. The following example shows a table schema:

```

+-----+
| Field | Type | Label | Comment |
+-----+
| id    | STRING | | |
| url   | STRING | | |
+-----+

```

- Input OSS file

Each line of the input OSS file is a URL or an OSS path, for example:

```

oss://your/path/to/image.jpg
http://your.path/to/image.jpg

```

- Custom input format

By default, only image URLs or Base64-encoded image data entries are read from the input tables. Only image URLs are read from the input OSS files to download and decode images. Both of the two methods generate only the NumPy arrays of images in the {"image": np.ndarray} format. The NumPy arrays are used by the process and predictor classes to make predictions. More and more users customize the predictor and process classes. A single input data format cannot satisfy user needs. To support custom formats, the method for reading data from OSS files is improved.

The custom format can be the original OSS file format or JSON string format. Each line in the input files is a JSON string. You can enter multiple key-value pairs. All key-value pairs are saved in a dictionary and pass to the custom predictor and process classes. You can obtain the corresponding values based on custom keys.

If the values are OSS paths or URLs, the system automatically uses multiple threads to download the file content, and saves the values in a Python file-like object. You can call methods such as `read()` or `readlines()` to obtain the content of the Python file-like object. If a value points to a file with an image extension, the system automatically decodes the image. You can obtain the value from the `input_data` dictionary based on the corresponding key. The value is of the `numpy.ndarray` type.

The following code provides an example of the input data:

```
{"image":"http://your/path/to/image.jpg", "prior":"oss://your/path/to/prior.txt", "config":{"key1":1, "key2":"value2"}}
{"image":"http://your/path/to/image.jpg", "prior":"oss://your/path/to/prior.txt", "config":{"key2":1, "key2":"value2"}}
```

The preceding input data will be converted into data in the `input_data` dictionary with the following fields:

- `image`: the decoded data of an image.
- `prior`: the file-like object.
- `config`: a dictionary of JSON strings.

The `input_data` dictionary is in the following format. For all custom process and predictor classes, you can query image data based on the key.

```
input_dict = {
    "image": np.ndarray,
    "prior": file_like_object,
    "config": {"key1":1, "key2":"value2"}
}
```

 **Note** All built-in predictor classes of EasyVision use the `image` key to obtain input images. If you want to use a custom input format to call the built-in predictor classes of EasyVision, you must set the `image` key for image data.

Custom third-party library

The offline prediction framework is implemented based on PAI-TensorFlow. Therefore, the framework does not allow you to use PyTorch models to make custom predictions. In addition, if you make a custom prediction by using a third-party library that does not exist in the preset Python environment, the prediction fails. To resolve this issue, Machine Learning Platform for AI allows you to use the `requirements.txt` file to install a custom Python library.

You must add the `requirements.txt` file to the top-level code directory:

```
user_resource
├── pytorch_sr_script.py
├── requirements.txt
├── user_predictor.py
└── user_process.py
```

Then, you must package the entire code directory and upload it to OSS. The following code provides an example:

```
tar -zcf user_resource.tar.gz user_resource
osscli put user_resource.tar.gz oss://your/path/user_resource.tar.gz
```

You can specify the address of the code package in the Machine Learning Platform for AI command parameter `-Duser_resource`. When the prediction task runs, Machine Learning Platform for AI automatically installs the dependencies that are specified in the `requirements.txt` file.

Custom predictor class

The custom predictor class must inherit `easy_vision.python.inference.predictor.PredictorInterface` or `easy_vision.python.inference.predictor.PredictorInterfaceV2`. The names and default values of all functions and parameters of the custom predictor class must be the same as those in the operation definition.

The general image prediction component is compatible with V1 and V2 operation definitions of the custom predictor class.

- V1 (only serializes all prediction results to JSON strings)

The following code shows the operation definition:

```
class PredictorInterface(six.with_metaclass(abc.ABCMeta, object)):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        pass
    @abc.abstractmethod
    def predict(self, input_data, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data: a list of numpy array, each array is a sample to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
                use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
                eg, {"output1": value1, "output2": value2}, the value type can be
                python int str float, and numpy array
        """
        pass
```

The following code provides an example of the custom predictor class:

```
#!/usr/bin/env python
# -*- encoding:utf-8 -*-
# Filename: user_predictor.py
# Author: Qianyan (wenmeng.zwm@alibaba-inc.com)
# Date: 2019-06-10
# Description:
from easy_vision.python.inference import predictor
import tensorflow as tf
class MyPredictor(predictor.PredictorInterface):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        tf.logging.info('my predictor init done')
    def predict(self, input_data, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data: a list of numpy array, each array is a sample to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
                use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
                eg, {"output1": value1, "output2": value2}, the value type can be
                python int str float, and numpy array
        """
        results = [{'result1': 'my_predictor_result1', 'result2': 'my_predictor_result2'}
                    for i in range(len(input_data))]
        return results
```

- V2

The custom predictor class has the following limits:

- You can use the custom predictor class to pass only the list of images. No other information can be passed.
- The output dictionary is converted into only JSON strings and cannot save image files.

To remove the limits, Machine Learning Platform for AI provides V2 operation definition to enable you to specify the output format. V2 is developed based on V1 by adding the `get_output_type` function. This function returns a dictionary that indicates how to save the values in the key-value pairs in the dictionary returned by the prediction. The values can be saved in the following ways:

- json: All the output JSON data of keys is aggregated into a dictionary and serialized to a JSON string. Then, the JSON string is written to the output OSS file.
- image: The output data of the keys that correspond to values of the image type is saved in an OSS file *output_dir/key/filename.jpg*. The value of the filename variable is obtained from the input URL. If the output data of a key is a list of images, the images are saved as *output_dir/key/filename_idx.jpg*, where, idx is the index number of each image in the list.
- video: not supported.

To pass more information, you can also pass a list of dictionaries to the predictor class by using custom input data types.

The following code shows the operation definition:

```
class PredictorInterfaceV2(six.with_metaclass(abc.ABCMeta, object)):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        pass
    def get_output_type(self):
        """
        in this function user should return a type dict, which indicates
        which type of data should the output of predictor be converted to
        * type json, data will be serialized to json str
        * type image, data will be converted to encode image binary and write to oss file,
        whose name is output_dir/{key}/{input_filename}_{idx}.jpg, where input_filename
        is extracted from url, key corresponds to the key in the dict of output_type,
        if the type of data indexed by key is a list, idx is the index of element in list, otherwise {idx} will be empty
        * type video, data will be converted to encode video binary and write to oss file,
        eg: return {
            'ret_image': 'image',
            'feature': 'json'
        }
        indicating that the image data in the output dict will be save to image
        file and feature in output dict will be converted to json
        """
        return {}
    @abc.abstractmethod
    def predict(self, input_data_dict, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data_dict: a list of dict, each dict is a sample data to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
            use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
            eg, {"output1": value1, "output2": value2}, the value type can be
            python int str float, and numpy array
        """
        pass
```

The following code provides an example of the custom predictor class:

```
class DummyMultiModalPredictor(ev_predictor.PredictorInterfaceV2):
    def __init__(self, model_path, model_config=None):
        pass
    def get_output_type(self):
        return {'result1': 'json',
                'result2': 'json',
                'result_image': 'image',
                'result_image_list': 'image',
                }
    def predict(self, images, batch_size=1):
        results = [
            {'result1': 'dummy',
             'result2': 'dummy',
             'result_image': np.zeros([224,224,3],dtype=np.uint8),
             'result_image_list': [255* np.ones([224,224,3],dtype=np.uint8) for i in range(2)]
            } for i in range(len(images))]
        return results
```

Custom process class

The custom process class must inherit `easy_vision.python.pai_utils.ev_process.UserProcess`. The names and default values of all functions and parameters of the custom process class must be the same as those in the operation definition.

The following code shows the operation definition:

```
class UserProcess(Process):
    def __init__(self,
                 job_name,
                 num_threads=1,
                 input_queue=None,
                 output_queue=None,
                 process_config=None,
                 batch_size=1):
        """
        Args:
        job_name: job name for this process
        num_threads: number of threads to run your process
        input_queue: a queue containing input data, you should not read input_queue by
        yourself
        output_queue: you can push data to output_queue or just return data in process
        which will be pushed into output queue automatically
        process_config: config for your process, dict type
        batch_size: if batch_size is 1, input_data passed to in_data is a dict of one sample data
        if batch_size greater than 1, input_data passed to process is a list of dict, each dict
        contains one sample data
        """
        super(UserProcess, self).__init__(
            job_name,
            num_threads,
            input_queue=input_queue,
            output_queue=output_queue,
            batch_size=batch_size)
    def process(self, in_data):
        """
        method need to be reimplemented, one can add result to output_queue in this func
        or just return the result which will be pushed to output queue automatically.
        you should add the process result to input_data dict, and push it to the output
        queue or just return it. You can refer to easy_vision.python.pai_utils.ev_process.DataFields
        attribute to acquire what information you can get from input_dict, you can only
        write data with key like DataFields.image or DataFields.clip
        Args:
        in_data if self.batch_size is 1, input_data is a dict of one sample data
        if self.batch_size greater than 1, input_data is a list of dict, each dict
        contains one sample data
        Return
        if None, result should be added to output_queue by yourself in this func
        if not None, the returned result will be add to output_queue automatically
        """
        return input_data
    def destroy(self):
        """
        destroy resources that has been used for this process
        """
        pass
```

The following code provides an example of the custom process class:

```
#!/usr/bin/env python
# -*- encoding:utf-8 -*-
# Filename: user_process.py
# Author: Qianyan (wenmeng.zwm@alibaba-inc.com)
# Date: 2019-06-10
```

```

# Date: 2019-08-16
# Description:
from easy_vision.python.pai_utils import ev_process
import logging
class MyProcess(ev_process.UserProcess):
    def __init__(self,
                 job_name,
                 num_threads=1,
                 input_queue=None,
                 output_queue=None,
                 process_config=None,
                 batch_size=1):
        """
        Args:
        job_name: job name for this process
        num_threads: number of threads to run your process
        input_queue: a queue containing input data, you should not read input_queue by
        yourself
        output_queue: you can push data to output_queue or just return data in process
        which will be pushed into output queue automatically
        process_config: config for your process, dict type
        batch_size: if batch_size is 1, input_data passed to in_data is a dict of one sample data
        if batch_size greater than 1, input_data passed to process is a list of dict, each dict
        contains one sample data
        """
        super(MyProcess, self).__init__(job_name, num_threads,
                                       input_queue = input_queue,
                                       output_queue = output_queue,
                                       batch_size = batch_size)
    def process(self, in_data):
        """
        method need to be reimplemented, one can add result to output_queue in this func
        or just return the result which will be pushed to output queue automatically.
        you should add the process result to input_data dict, and push it to the output
        queue or just return it. You can refer to easy_vision.python.pai_utils.ev_process.DataFields
        attribute to acquire what information you can get from input_dict, you can only
        write data with key like DataFields.image or DataFields.clip
        Args:
        in_data if self.batch_size is 1, input_data is a dict of one sample data
        if self.batch_size greater than 1, input_data is a list of dict, each dict
        contains one sample data
        Return
        if None, result should be added to output_queue by yourself in this func
        if not None, the returned result will be add to output_queue automatically
        """
        logging.info('my process process data')
        return in_data
    def destroy(self):
        """
        destroy resources that has been used for this process
        """
        logging.info('my process destroyed')

```

Result format

This section describes the result format of various models.

- feature_extractor

Sample result :

```
{"feature": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4583122730255127, 0.0]}
```

The following table describes the field in the sample result.

Field	Description	Shape	Type
feature	The extracted feature.	[feature_dim]	FLOAT

- classifier

Sample result :

```
{  
  "class": 3,  
  "class_name": "coho4",  
  "class_probs": {"coho1": 4.028851974258174e-10,  
    "coho2": 0.48115724325180054,  
    "coho3": 5.116515922054532e-07,  
    "coho4": 0.5188422446937221}  
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
class	The ID of the category.	[]	INT 32
class_name	The name of the category.	[]	STRING
class_probs	The matching probabilities of all categories.	[num_classes]	DICT{key: STRING, value: FLOAT}

- detector

Sample result :

```
{  
  "detection_boxes": [[243.5308074951172, 197.69570922851562, 385.59625244140625, 247.7247772216797], [292.1929931640625, 114.28043365478516, 571.2748413085938, 165.09771728515625]],  
  "detection_scores": [0.9942291975021362, 0.9940272569656372],  
  "detection_classes": [1, 1],  
  "detection_class_names": ["text", "text"]  
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
detection_boxes	The text area with detected coordinates in the following order: [top, left, bottom, right].	[num_detections, 4]	FLOAT

Field	Description	Shape	Type
detection_scores	The detection probability of the object.	num_detections	FLOAT
detection_classes	The ID of the category to which the object belongs.	num_detections	INT
detection_class_names	The name of the category to which the object belongs.	num_detections	ISTRING
detection_masks	The segmentation masks of the object. This parameter is optional.	[num_detection, image_height, image_width]	FLOAT
detection_keypoints	The landmarks of the object. This parameter is optional.	[num_detection, num_keypoints, 2]	FLOAT
detection_roi_features	The local features of the object. This parameter is optional.	[num_detection, roi_height, roi_width, channels]	FLOAT

- segmentor

Sample result :

```
{
  "probs" : [[[0.8, 0.8], [0.6, 0.7]], [[0.8, 0.5], [0.4, 0.3]]],
  "preds" : [[[1,1], [0, 0]], [[0, 0], [1,1]]]
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
probs	The probability that each pixel obtained through segmentation belongs to a specific category.	[output_height, output_width, num_classes]	FLOAT
preds	The ID of the category to which the pixel obtained through segmentation belongs.	[output_height, output_widths]	INT

- text_detector

Sample result :

```
{
  "detection_keypoints": [[[243.57516479492188, 198.84210205078125], [243.91038513183594, 247.62425231933594], [385.5513916015625, 246.61660766601562], [385.2197570800781, 197.79345703125]], [[292.2718200683594, 114.44700622558594], [292.2237243652344, 164.684814453125], [571.1962890625, 164.931640625], [571.2444458007812, 114.67433166503906]]],
  "detection_boxes": [[243.5308074951172, 197.69570922851562, 385.59625244140625, 247.7247772216797], [292.1929931640625, 114.28043365478516, 571.2748413085938, 165.09771728515625]],
  "detection_scores": [0.9942291975021362, 0.9940272569656372],
  "detection_classes": [1, 1],
  "detection_class_names": ["text", "text"],
  "image_shape": [1024, 968, 3]
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
detection_boxes	The detected text area with coordinates in the following order: [top, left, bottom, right].	[num_detections, 4]	FLOAT
detection_scores	The probability that the text is detected.	num_detections	FLOAT
detection_classes	The ID of the category to which the text area belongs.	num_detections	INT
detection_class_names	The name of the category to which the text area belongs.	num_detections	STRING
detection_keypoints	The (y, x) coordinates of the four vertices of the detected text area.	[num_detections, 4, 2]	FLOAT
image_shape	The information about the input image. The value is a list that contains the following information in order: height, width, and channel.	[3]	LIST

- text_recognizer

Sample result :

```
{
  "sequence_predict_ids": [1,2,2008,12],
  "sequence_predict_texts": "This is an example",
  "sequence_probability": 0.88
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
sequence_predict_ids	The ID of the category to which a single line of the recognized text belongs.	[text_length]	INT
sequence_predict_texts	The recognition result of the single-line text.	[]	STRING
sequence_probability	The probability that the single-line text is recognized.	[]	FLOAT

- text_spotter

Sample result :

```
{
  "detection_keypoints": [[[243.57516479492188, 198.84210205078125], [243.91038513183594, 247.62425231933594], [385.5513916015625, 246.61660766601562], [385.2197570800781, 197.79345703125]], [[292.2718200683594, 114.44700622558594], [292.2237243652344, 164.684814453125], [571.1962890625, 164.931640625], [571.2444458007812, 114.67433166503906]]],
  "detection_boxes": [[243.5308074951172, 197.69570922851562, 385.59625244140625, 247.7247772216797], [292.1929931640625, 114.28043365478516, 571.2748413085938, 165.09771728515625]],
  "detection_scores": [0.9942291975021362, 0.9940272569656372],
  "detection_classes": [1, 1],
  "detection_class_names": ["text", "text"],
  "detection_texts_ids": [[1,2,2008,12], [1,2,2008,12]],
  "detection_texts": ["This is an example", "This is an example"],
  "detection_texts_scores": [0.88, 0.88],
  "image_shape": [1024, 968, 3]
}
```

The following table describes the fields in the sample result.

Field	Description	Shape	Type
detection_boxes	The detected text area with coordinates in the following order: [top, left, bottom, right].	[num_detections, 4]	FLOAT
detection_scores	The probability that the text is detected.	num_detections	FLOAT
detection_classes	The ID of the category to which the text area belongs.	num_detections	INT
detection_class_names	The name of the category to which the text area belongs.	num_detections	STRING
detection_keypoints	The (y, x) coordinates of the four vertices of the detected text area.	[num_detections, 4, 2]	FLOAT

Field	Description	Shape	Type
detection_texts_ids	The ID of the category to which a single line of the recognized text belongs.	[num_detections, max_text_length]	INT
detection_texts	The recognition result of the single-line text.	[num_detections]	STRING
detection_texts_scores	The probability that the single-line text is recognized.	[num_detections]	FLOAT
image_shape	The information about the input image. The value is a list that contains the following information in order: height, width, and channel.	[3]	LIST

- Custom result format

By default, `ev_predict` extracts the dictionary from the cache queue, obtains the value of `prediction_result`, serializes the value to a JSON string, and then writes the JSON string to the result column. To write results to different columns by field, you can set the `self_defined_result_formatter` parameter to the path of the custom result formatting class and the `user_resource` parameter to the path of your code.

The process class is a derived class. For more information about the operation definition of this class, see the "Custom process class" section of this topic. You need only to save the formatted results in the `input_data` dictionary with the key as `result_to_save`. Then, `TableWriter` automatically writes the prediction results to the specified table.

The following code provides an example on how to write the decoded data of a video to a table by frame. Each line is the encoded data of a frame.

```

class SelfDefineTableFormatProcess(ev_process.Process):
    def __init__(self,
                 input_queue,
                 output_queue,
                 reserved_col_names=[],
                 output_col_names=[]):
        """
        extract data from dict to prepare result dict which will be written to table or oss file
        Args:
            reserved_col_names list of str column names for reserved col
            output_col_names list of column name for output table
            input_queue the python queue for input
            output the python queue for output
            float_digits: The number of digits of precision when writing floats out
        Return:
            push a k-v pair {'result_to_save': result_dict} into data_dict and return
            a result dict or a tuple, or a list of tuple or a list of dict
            if dict is returned, each key should be the same as column name of output table
            if tuple is returned, this tuple correspond to one table record
            if list of tuple is returned, they correspond to a list of table record
            if list of dict is returned, each element in list will be automatically converted to table record in table writer
        """
        super(SelfDefineTableFormatProcess, self).__init__('SelfDefineTableFormatProcess', 1,
                                                         input_queue=input_queue,
                                                         output_queue=output_queue)
    def process(self, input_data):
        output_record = []
        frames = input_data['frames']
        frames = [base64.urlsafe_b64encode(np.array(frame)) for frame in frames]
        frame_ids = input_data['frame_id']
        url = input_data['url']
        video_records = []
        for url, fid, frame in zip(url, frame_ids, frames):
            video_records.append((url, fid, frame))
        # result should be returned with key result_to_save
        input_data['result_to_save'] = video_records
        return input_data

```

Add the preceding code to `oss://path/to/you/code.py`. Then, use the following Machine Learning Platform for AI command parameters to customize the logic for writing data to the table.

```

pai -name ev_predict
...
-Duser_resource=oss://path/to/you/code.py
-Dself_defined_result_formatter=code.SelfDefineTableFormatProcess

```

4.4.13.3.2. General video prediction

This topic provides examples on how to run a Machine Learning Platform for AI command to use the video prediction component. This topic also describes the command parameters and provides the operation definitions of and examples on custom process and predictor classes.

You can run the `ev_video_predict` command to make video predictions. When you run this command, you need only to set the common parameters without the need to know the rules or logic of the EasyVision profile.

Machine Learning Platform for AI command example

- Input OSS file

```
pai -name ev_video_predict
  -Dinput_oss_file='oss://pai-vision-data-hz/data/pai_test/video_decode_test.list'
  -Doutput_oss_file='oss://pai-vision-data-hz/test/tmp/ev_video_predict_result.txt'
  -Dmodel_path='oss://pai-vision-data-hz/data/pai_test/ev_predict/test_data/models/ucf101_resnet3d_50'
  -Dtask_type='video_classifier'
  -Dfeature_name=''
  -Ddecode_type=4
  -Dsample_fps=-1
  -Dreshape_size=-1
  -Dsample_duration=16
  -Dbatch_size=16
  -Dpredict_thread_num=1
  -Dqueue_size=256
  -Dnum_worker=4
  -DcpuRequired=400
  -Dbuckets='YOUR_OSS_BUCKET_CONFIG'
```

- Input Swift file

```
pai -name ev_video_predict
  -Dswift_end_point='SWIFT_END_POINT'
  -Dinput_topic='INPUT_TOPIC'
  -Doutput_topic='OUTPUT_TOPIC'
  -Dmodel_path='oss://pai-vision-data-hz/data/pai_test/ev_predict/test_data/models/ucf101_resnet3d_50'
  -Dtask_type='video_classifier'
  -Dfeature_name=''
  -Ddecode_type=4
  -Dsample_fps=-1
  -Dreshape_size=-1
  -Dsample_duration=16
  -Dbatch_size=16
  -Dpredict_thread_num=1
  -Dqueue_size=256
  -Dnum_worker=4
  -DcpuRequired=400
  -Dbuckets='YOUR_OSS_BUCKET_CONFIG'
```

- Custom process and predictor classes

```

pai -name ev_video_predict
-Dswift_end_point='SWIFT_END_POINT'
-Dinput_topic='INPUT_TOPIC'
-Doutput_topic='OUTPUT_TOPIC'
-Dmodel_path='oss://pai-vision-data-hz/data/pai_test/ev_predict/test_data/models/ucf101_resnet3d_50'
-Dtask_type='self_define'
-Dfeature_name=''
-Ddecode_type=4
-Dsample_fps=-1
-Dreshape_size=-1
-Dsample_duration=16
-Dbatch_size=1
-Dpredict_thread_num=1
-Dqueue_size=256
-Dnum_worker=4
-DcpuRequired=400
-Duser_resource='oss://pai-vision-data-hz/data/pai_test/ev_predict/test_data/self_define/video_res.tar.gz'
-Duser_predictor_cls='user_predictor.MyPredictor'
-Duser_process_config='
[
{
  "job_name": "myprocess",
  "num_threads": 4,
  "batch_size": 1,
  "class": "user_process.MyProcess"
}
]'
-Dbuckets='oss://pai-vision-data-hz/?role_arn=acs:ram::121706069718XXXX:role/pai-vision-hz&host=oss-cn-zhangjiakou.aliyuncs.com'

```

The code file that is used to customize the predictor class is *user_predictor.py*. The class name is `MvPredictor`. The code file that is used to customize the process class is *user_process.py*. The class name is `MyProcess`. The *user_predictor.py* and *user_process.py* code files are compressed in the *video_res.tar.gz* package.

Parameters

Parameter	Required	Description	Value type or value format	Default value
buckets	Yes	The information about the OSS buckets.	"oss://bucket_name/?role_arn=xxx&host=yyy"	N/A
model_path	Yes	The OSS path of the model.	"oss://your_bucket/your_model_dir" The OSS path of the model that EasyVision provides is <code>oss://pai-vision-data-hz/pretrained_models/saved_models/ucf101_resnet3d_50/</code> .	N/A

Parameter	Required	Description	Value type or value format	Default value
task_type	Yes	The type of the model. Only the video classification model is supported. Set this parameter to video_classifier.	STRING	N/A
input_oss_file	Yes	The path of the input OSS file. Each line in the input file can be in one of the following formats: <ul style="list-style-type: none"> The path of a video. A JSON string. Example: {"video_binary_data": VIDEO_URL, "xxx":xxx}. 	oss://your_bucket/filelist.txt	N/A
output_oss_file	Yes	The path of the output OSS file that is used to store the prediction results. The system may generate multiple result files and merge these result files into an output OSS file. The result files are prefixed with the name of the output OSS file. The number of result files is the same as the number of workers, which is specified by the num_worker parameter.	oss://your_bucket/result.txt	N/A
output_dir	No	The directory of the output file. If you customize the output format, all the result files are stored in this directory.	oss://your_bucket/dir	""
swift_end_point	No	The URL of the online ZooKeeper server.	STRING	""
input_topic	No	The input topic. Each element in the topic is the OSS path of a video or a JSON string, for example, {"url":xxx, "user_define": xxx}.	STRING	""
output_topic	No	The output topic.	STRING	""
feature_name	No	The name of the output feature. Example: "resnet_3d_50/poo2", "resnet_3d_50/scale5/conv5", "resnet_3d_50/scale4/conv4", "resnet_3d_50/scale3/conv3", "resnet_3d_50/scale2/conv2", "resnet_3d_50/pool1", "resnet_3d_50/scale1/conv1".	STRING	""

Parameter	Required	Description	Value type or value format	Default value
decode_type	No	The method that is used to decode the videos. Valid values: <ul style="list-style-type: none"> • 1: intra only • 2: keyframe only • 3: without bidir • 4: decode all 	INT	4
sample_fps	No	The number of frames that are extracted for sampling per second.	FLOAT	-1, which indicates that 25 frames are extracted per second.
reshape_size	No	The size of the output frames, in pixels.	INT	If you set this parameter to -1, the frames are not resized.
sample_duration	No	The length of consecutive frames that are imported to the model.	INT	16
batch_size	No	The number of clips that are imported to the model.	INT	16
predict_thread_num	No	The number of threads that are used for prediction.	INT	1
queue_size	No	The length of the processing queue.	INT	256
num_worker	No	The number of prediction workers.	INT	2
cpuRequired	No	The number of CPUs for a worker. A value of 100 indicates one CPU.	INT	400
gpuRequired	No	The number of GPUs for a worker. A value of 100 indicates one GPU.	INT	100
user_resource	No	The path to which you want to upload your resources. You can upload tar.gz, .zip, and .py files. OSS paths and HTTP URLs are supported.	oss://xxx/a.tar.gz http://a.com/c.zip	""

Parameter	Required	Description	Value type or value format	Default value
user_predictor_cls	No	The module path of the custom predictor class. For example, if you implement Predictor A in module.py, the module path of Predictor A is module.A.	module.PredictorA	""
user_process_config	No	<p>The configurations of the custom process class, in the JSON string format. You can use the following fields to configure the process class. You can also configure other custom fields.</p> <ul style="list-style-type: none"> • job_name: the name of the custom process class. • num_threads: the number of concurrent threads that are used by the custom process class. • batch_size: the batch size of the data to be processed. • user_process_cls: the module path of the custom process class. For example, if you implement Process A in module.py, the module path of Predictor A is module.A. 	<pre>{["job_name": "myprocess", "user_process_cls": "module.ClassA", "num_threads": 2, "batch_size": 1]}</pre>	""

Result format

The result of each video is a JSON string, which includes the URL or OSS path and the prediction result of the video. The following code provides an example:

```

oss://pai-vision-data-hz/data/ucf101_train/ApplyEyeMakeup/v_ApplyEyeMakeup_g08_c01,{"class_name": "ApplyEyeMakeup", "class": 0, "probs": {"MilitaryParade": 6.785883670090698e-06, "TrampolineJumping": 1.9762439933401765e-06, "PlayingDaf": 2.7828968086396344e-05, "SalsaSpin": 7.392855422949651e-06, "CuttingInKitchen": 2.6925881684292108e-05, "ApplyEyeMakeup": 0.9972793459892273, "PlayingViolin": 1.8471011571818963e-05, "YoYo": 6.056292932044016e-06, "PlayingDhol": 4.430206445249496e-06, "PlayingCello": 2.7006941309082322e-05, "Bowling": 7.969072612468153e-06, "UnevenBars": 3.238687213524827e-06, "BalanceBeam": 5.145544491824694e-06, "SkyDiving": 3.6508849916572217e-06, "SumoWrestling": 9.847853107203264e-06, "PushUps": 9.45046849665232e-06, "FloorGymnastics": 3.89273827750003e-06, "ApplyLipstick": 0.0008350351126864552, "BreastStroke": 2.5186193397530587e-06, "GolfSwing": 2.3900222458905773e-06, "HorseRiding": 3.5062098504567984e-06, "PlayingFlute": 1.801957296265755e-05, "PizzaTossing": 7.031375389487948e-06, "CleanAndJerk": 4.537632776191458e-06, "WritingOnBoard": 2.856705032172613e-05, "CricketShot": 4.059567345393589e-06, "FieldHockeyPenalty": 4.859361069975421e-06, "HammerThrow": 4.741576958622318e-06, "BodyWeightSquats": 5.07304139318876e-06, "CliffDiving": 7.602381629112642e-06, "Typing": 6.037013008608483e-05, "MoppingFloor": 7.247148005262716e-06, "TaiChi": 7.273819846886909e-06, "PlayingPiano": 9.108782251132652e-06, "Punch": 1.3467762983054854e-05, "Nunchucks": 7.87033604865428e-06, "RopeClimbing": 4.0648433241585735e-06, "Swing": 2.3137952211982338e-06, "Knitting": 2.0819035853492096e-05, "Rafting": 2.7667874746839516e-06, "PlayingGuitar": 2.7430540285422467e-05, "VolleyballSpiking": 4.066964265803108e-06, "ShavingBeard": 0.0002367567940382287, "JugglingBalls": 1.696763138170354e-05, "Diving": 5.02031662108493e-06, "JumpingJack": 1.1237583748879842e-05, "PoleVault": 3.3857772905321326e-06, "SkateBoarding": 7.812836884113494e-06, "BoxingPunchingBag": 6.725965249643195e-06, "IceDancing": 1.026979043672327e-05, "WallPushups": 2.3817137844162062e-05, "FrisbeeCatch": 7.734954124316573e-06, "Drumming": 4.163131961831823e-06, "JumpRope": 9.629309715819545e-06, "HeadMassage": 6.168564868858084e-05, "PlayingTabla": 2.9489216103684157e-05, "TableTennisShot": 6.383983418345451e-06, "PommelHorse": 5.876625891687581e-06, "HighJump": 3.0327462354762247e-06, "BasketballDunk": 5.714619419450173e-06, "BoxingSpeedBag": 2.9891823942307383e-05, "PullUps": 7.257571269292384e-06, "SoccerPenalty": 5.835005140397698e-06, "RockClimbingIndoor": 9.057837814907543e-06, "BlowingCandles": 5.086950750410324e-06, "Skiing": 6.789191957068397e-06, "WalkingWithDog": 3.2428008580609458e-06, "Basketball": 4.267759322829079e-06, "SoccerJuggling": 4.374186573841143e-06, "Fencing": 6.132470389275113e-06, "Billiards": 5.670899554388598e-06, "BaseballPitch": 4.975987394573167e-06, "BlowDryHair": 0.0001697591069387272, "CricketBowling": 2.135262002411764e-05, "BandMarching": 4.6766035666223615e-06, "PlayingSitar": 3.0760954814468278e-06, "ThrowDiscus": 1.1850976079585962e-05, "StillRings": 5.105009677208727e-06, "Lunges": 1.5529138863712433e-06, "Skijet": 3.188117261743173e-06, "BabyCrawling": 5.77771561438567e-06, "Mixing": 4.743058525491506e-05, "Hammering": 7.294750957953511e-06, "Shotput": 3.6980163713451475e-06, "Archery": 4.3028112486354075e-06, "Surfing": 5.78391745875706e-06, "FrontCrawl": 8.995367352326866e-06, "HulaHoop": 9.055997907125857e-06, "JavelinThrow": 8.379415703529958e-06, "Rowing": 6.835780368419364e-06, "Kayaking": 3.7107411117176525e-06, "ParallelBars": 6.430962002923479e-06, "HorseRace": 9.8691098173731e-06, "HandstandWalking": 4.0894760786613915e-06, "BrushingTeeth": 0.0001521828380646184, "LongJump": 1.0337393177906051e-05, "Biking": 5.364606295188423e-06, "HandstandPushups": 6.265170213737292e-06, "BenchPress": 1.0012458915298339e-05, "Haircut": 0.0003677888307720423, "TennisSwing": 3.352387921040645e-06}}

```

If you want to obtain video features, the result contains the feature field with the corresponding feature value. The following code provides an example:

```
{"feature": [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4583122730255127, 0.0]}
```

Parameter	Description	Shape	Value type
class	The ID of the category.	[]	INT32
class_name	The name of the category.	[]	STRING
probs	The matching probabilities of all categories.	[num_classes]	DICT {Key: STRING, Value: FLOAT}
feature	The video feature.	[]	FLOAT

Custom process class

The custom process class must inherit `easy_vision.python.pai_utils.ev_process.UserProcess`. The names and default values of all functions and parameters of the custom process class must be the same as those in the operation definition.

The following code shows the operation definition:

```
class UserProcess(Process):
    def __init__(self,
                 job_name,
                 num_threads=1,
                 input_queue=None,
                 output_queue=None,
                 process_config=None,
                 batch_size=1):
        """
        Args:
            job_name: job name for this process
            num_threads: number of threads to run your process
            input_queue: a queue containing input data, you should not read input_queue by
            yourself
            output_queue: you can push data to output_queue or just return data in process
            which will be pushed into output queue automatically
            process_config: config for your process, dict type
            batch_size: if batch_size is 1, input_data passed to in_data is a dict of one sample data
            if batch_size greater than 1, input_data passed to process is a list of dict, each dict
            contains one sample data
        """
        super(UserProcess, self).__init__(
            job_name,
            num_threads,
            input_queue=input_queue,
            output_queue=output_queue,
            batch_size=batch_size)
    def process(self, in_data):
        """
        method need to be reimplemented, one can add result to output_queue in this func
        or just return the result which will be pushed to output queue automatically.
        you should add the process result to input_data dict, and push it to the output
        queue or just return it. You can refer to easy_vision.python.pai_utils.ev_process.DataFields
        attribute to acquire what information you can get from input_dict, you can only
        write data with key like DataFields.image or DataFields.clip
        Args:
            in_data if self.batch_size is 1, input_data is a dict of one sample data
            if self.batch_size greater than 1, input_data is a list of dict, each dict
            contains one sample data
        Return
            if None, result should be added to output_queue by yourself in this func
            if not None, the returned result will be add to output_queue automatically
        """
        return in_data
    def destroy(self):
        """
        destroy resources that has been used for this process
        """
        pass
```

To use the custom process class for video prediction, you must pass a dictionary that contains the following fields:

- `url`: the URL of the video.
- `video_binary_data`: the original video stream, in the `STRING` format.
- `frames`: the frames that are decoded from the video, in the `LIST` format. In the list, each element indicates a frame in the `STRING` format.
- `time_stamp`: the timestamps, in the `LIST` format. In the list, each element indicates a timestamp in the `FLOAT` format.
- `xxx`: the custom information.

The following code provides an example of the custom process class:

```
#!/usr/bin/env python
# -*- encoding:utf-8 -*-
# Filename: user_process.py
from easy_vision.python.pai_utils import ev_process
import logging
class MyProcess(ev_process.UserProcess):
    def __init__(self,
                 job_name,
                 num_threads=1,
                 input_queue=None,
                 output_queue=None,
                 process_config=None,
                 batch_size=1):
        """
        Args:
        job_name: job name for this process
        num_threads: number of threads to run your process
        input_queue: a queue containing input data, you should not read input_queue by
        yourself
        output_queue: you can push data to output_queue or just return data in process
        which will be pushed into output queue automatically
        process_config: config for your process, dict type
        batch_size: if batch_size is 1, input_data passed to in_data is a dict of one sample data
        if batch_size greater than 1, input_data passed to process is a list of dict, each dict
        contains one sample data
        """
        super(MyProcess, self).__init__(job_name, num_threads,
                                       input_queue = input_queue,
                                       output_queue = output_queue,
                                       batch_size = batch_size)
    def process(self, in_data):
        """
        method need to be reimplemented, one can add result to output_queue in this func
        or just return the result which will be pushed to output queue automatically.
        you should add the process result to input_data dict, and push it to the output
        queue or just return it. You can refer to easy_vision.python.pai_utils.ev_process.DataFields
        attribute to acquire what information you can get from input_dict, you can only
        write data with key like DataFields.image or DataFields.clip
        Args:
        in_data if self.batch_size is 1, input_data is a dict of one sample data
        if self.batch_size greater than 1, input_data is a list of dict, each dict
        contains one sample data
        Return
        if None, result should be added to output_queue by yourself in this func
        if not None, the returned result will be add to output_queue automatically
        """
        logging.info('my process process data')
        return in_data
    def destroy(self):
        """
        destroy resources that has been used for this process
        """
        logging.info('my process destroyed')
```

Custom predictor class

The custom predictor class must inherit `easy_vision.python.inference.predictor.PredictorInterface`. The names and default values of all functions and parameters of the custom predictor class must be the same as those in the operation definition.

The general video prediction component is compatible with V1 and V2 operation definitions of the custom predictor class.

- V1

The following code shows the operation definition:

```
class PredictorInterface(six.with_metaclass(abc.ABCMeta, object)):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        pass
    @abc.abstractmethod
    def predict(self, input_data, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data: a list of numpy array, each array is a sample to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
                use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
                eg, {"output1": value1, "output2": value2}, the value type can be
                python int str float, and numpy array
        """
        pass
```

The following code provides an example of the custom predictor class:

```
#!/usr/bin/env python
# -*- encoding:utf-8 -*-
# Filename: user_predictor.py
# Description:
from easy_vision.python.inference import predictor
import tensorflow as tf
class MyPredictor(predictor.PredictorInterface):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        tf.logging.info('my predictor init done')
    def predict(self, input_data, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data: a list of numpy array[*h*w*c], each array is a sample to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
                use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
                eg, {"output1": value1, "output2": value2}, the value type can be
                python int str float, and numpy array
        """
        results = []
        for i in range(len(input_data)):
            output = {'result1':'my_predictor_result1'}
            if 'url' in input_data:
                output['url'] = input_data['url']
            if 'user_define' in input_data:
                output['user_define'] = input_data['user_define']
            results.append(output)
        return results
```

- V2

The custom predictor class has the following limits:

- You can use the custom predictor class to pass only the list of images. No other information can be passed.
- The output dictionary is converted into only JSON strings and cannot save image files.

To remove the limits, Machine Learning Platform for AI provides V2 operation definition to enable you to specify the output format. V2 is developed based on V1 by adding the `get_output_type` function. This function returns a dictionary that indicates how to save the values in the key-value pairs in the dictionary returned by the prediction. The values can be saved in the following ways:

- json: All the output JSON data of keys is aggregated into a dictionary and serialized to a JSON string. Then, the JSON string is written to the output OSS file.
- image: The output data of the keys that correspond to values of the image type is saved in an OSS file *output_dir/key/filename.jpg*. The value of the filename variable is obtained from the input URL. If the output data of a key is a list of images, each image in the list is saved as *output_dir/key/filename_idx.jpg*, where, *idx* is the index number of each image.
- video: not supported.

To pass more information, you can also pass a list of dictionaries to the predictor class by using custom input data types.

The following code shows the operation definition:

```
class PredictorInterfaceV2(six.with_metaclass(abc.ABCMeta, object)):
    def __init__(self, model_path, model_config=None):
        """
        init tensorflow session and load tf model
        Args:
            model_path init model from this directory
            model_config config string for model to init, in json format
        """
        pass
    def get_output_type(self):
        """
        in this function user should return a type dict, which indicates
        which type of data should the output of predictor be converted to
        * type json, data will be serialized to json str
        * type image, data will be converted to encode image binary and write to oss file,
        whose name is output_dir/${key}/${input_filename}_${idx}.jpg, where input_filename
        is extracted from url, key corresponds to the key in the dict of output_type,
        if the type of data indexed by key is a list, idx is the index of element in list, otherwise ${idx} will be empty
        * type video, data will be converted to encode video binary and write to oss file,
        eg: return {
            'image': 'jpg',
            'feature': 'json'
        }
        indicating that the image data in the output dict will be save to image
        file and feature in output dict will be converted to json
        """
        return {}
    @abc.abstractmethod
    def predict(self, input_data_dict, batch_size):
        """
        using session run predict a number of samples using batch_size
        Args:
            input_data_dict: a list of dict, each dict is a sample data to be predicted
            batch_size: batch_size passed by the caller, you can also ignore this param and
            use a fixed number if you do not want to adjust batch_size in runtime
        Return:
            result: a list of dict, each dict is the prediction result of one sample
            eg, {"output1": value1, "output2": value2}, the value type can be
            python int str float, and numpy array
        """
        pass
```

The following code provides an example of the custom predictor class:

```
class DummyMultiModalPredictor(ev_predictor.PredictorInterfaceV2):
    def __init__(self, model_path, model_config=None):
        pass
    def get_output_type(self):
        return {'result1': 'json',
                'result2': 'json',
                'result_image': 'image',
                'result_image_list': 'image',
                }
    def predict(self, images, batch_size=1):
        results = [
            {'result1': 'dummy',
             'result2': 'dummy',
             'result_image': np.zeros([224,224,3], dtype=np.uint8),
             'result_image_list': [255* np.ones([224,224,3], dtype=np.uint8) for i in range(2)]
            } for i in range(len(images))]
        return results
```

4.4.13.3.3. ASR prediction

This topic provides an example on how to run a Machine Learning Platform for AI command to use the automatic speech recognition (ASR) prediction component. This topic also describes the command parameters.

You must specify a MaxCompute table that contains the URL of the raw data as the input. Then, the ASR prediction component returns all the input data with an extra row at the end that contains the ASR result.

 **Note** You must create an empty table in advance as the output table. The schema of the output table must be the same as that of the input table. The last row of the output table contains the prediction result in the STRING format.

Machine Learning Platform for AI command example

```
PAI -name asr_predict_180 -project sre_mpi_algo_dev
-Dbuckets='oss://pai-audio-zjk/?role_arn=xxx&host=yyy'
-Dinput_tables='odps://sre_mpi_algo_dev/tables/video_asr_dev'
-Doutput_tables='odps://sre_mpi_algo_dev/tables/video_asr_dev_output'
-Durl_col_index=0
-Dmerge=1
-Dnum_workers=1
-Dnum_processes=6
-Dnum_download=7
-Dnum_memory=10000
-Doversubscription=false;
```

Set the buckets, input_tables, and output_tables parameters as needed.

Parameters

Parameter	Required	Description	Value type	Value
-----------	----------	-------------	------------	-------

Parameter	Required	Description	Value type	Value
buckets	Yes	The information about the Object Storage Service (OSS) buckets. Set this parameter to the path of a directory in an OSS bucket. The value format is the same as that of the buckets parameter of PAI-TensorFlow. Example: oss://pai-audio-zjk/?role_arn=xxx&host=yyy.	STRING	N/A
input_tables	Yes	The input tables. Example: odps://sre_mpi_algo_dev/tables/video_asr_dev.	STRING	N/A
output_tables	Yes	The output tables. Example: odps://sre_mpi_algo_dev/tables/video_asr_dev_output.	STRING	N/A
url_col_index	No	The serial number of the column that contains the data URL. The serial number starts from 0.	INT	0
merge	No	Specifies whether to return the merged recognition result. If you set this parameter to 0, the recognition result contains the speech segments and their start and stop timestamps. Example: {"detection_texts": ["你看人家想妈了吧小妈了", "小妈了", "太好了", "你呀赶紧给你妈打电话"], "time": [[["0", "2660"], ["4040", "4660"], ["5800", "6560"], ["6840", "10069"]]]}.	INT	1
num_workers	No	The number of workers that are used for prediction.	INT	1
num_processes	No	The number of threads that are used by each worker for prediction. We recommend that you set this parameter to 12 or less.	INT	6
num_download	No	The number of threads that are used by each worker to download videos. We recommend that you set this parameter to 10 or less.	INT	7
num_memory	No	The size of the memory that each worker consumes.	INT	10000

Parameter	Required	Description	Value type	Value
oversubscription	No	The value format of this parameter is the same as that of the oversubscription parameter of PAI-TensorFlow. <ul style="list-style-type: none"> true false 	BOOL	true

If an error occurs during ASR, the system returns the corresponding error code as a string in the output table. The following table describes the error codes.

Error code	Description
data_convert_error	An error occurred during data download or transcoding. Check whether the data source is normal.
data_download_error	An error occurred during data download. Check whether the URL is valid.
predict_error	An error occurred during the model prediction.

4.5. OpenAPI

4.5.1. Query PMML models

Operation name

ListPMMLModels

Description

You can call this operation to query PMML models by project ID, experiment ID, or owner.

Request parameters

Parameter	Type	Required	Example	Description
OnwerId	String	Yes	11769368777159105	The UID of a model owner.
ProjectId	Long	Yes	10009	The ID of a project.
ExperimentId	Long	No	4294	The ID of an experiment.
Action	String	Yes	ListPMMLModels	The operation that you want to perform. Set the value to ListPMMLModels.

Response parameters

Parameter	Type	Description
Experiments	List<ExperimentModelInfo>	An array of experiment properties returned.
Experiment	ExperimentModelInfo	The returned experiment information.
ExpId	Long	The ID of the experiment to which the models belong.
Models	List<ModelInfo >	An array of model properties returned.
Model	ModelInfo	The returned model information.
Name	String	The display name of the model.
Owner	String	The ID of the model owner.
Description	String	The description of the model.
ModelId	String	The ID of the model.
ModelName	String	The name of the underlying model generated by the corresponding algorithm.
CreateTime	Date	The time when the model was generated.
ExperimentId	String	The ID of the experiment from which the model is generated.
UpdateTime	Date	The time when the model was updated.
Project	String	The name of the project to which the model belongs.
ProjectId	String	The ID of the project to which the model belongs.
Action	String	The name of the API operation.
AccessKeyId	String	The AccessKey ID provided to you by Alibaba Cloud.
Signature	String	The signature string.
SignatureMethod	String	The signing method.
SignatureVersion	String	The version of the signature encryption algorithm.
SignatureNonce	String	A unique, random number used to prevent replay attacks.

Timestamp	String	The timestamp of the request.
Version	String	The version number of the API. The value must be in the YYYY-MM-DD format.
Format	String	The language of the response.

Sample requests

```
http://pop.pai.idst.inter.env8d.com/?Action=ListPMMLModels&ProjectId=10009&Version=2019-09-25&ExperimentId=4294&OnwerId=11769368777159105&<Common request parameters>
```

Sample responses

```
<ListPMMLModelsResponse>
<Experiments>
<Experiment>
<ExpId>4294</ExpId>
<Models>
<Model>
<Name>Logistic regression for binary classification-1-Model</Name>
<Owner>11769368777159105</Owner>
<Description>Logistic regression for binary classification-1-Model</Description>
<ModelId>6119</ModelId>
<ModelName>xlab_m_logisticregressi_51466_v0</ModelName>
<CreateTime>2019-09-23 17:06:42</CreateTime>
<ExperimentId>4294</ExperimentId>
<UpdateTime>2019-09-26 19:33:03</UpdateTime>
<Project>pai_emr</Project>
<ProjectId>10009</ProjectId>
</Model>
<Model>
<Name>Random forest1-AUC-0</Name>
<Owner>11769368777159105</Owner>
<Description>Random forest1-AUC-0</Description>
<ModelId>6200</ModelId>
<ModelName>xlab_m_random_forests_1_51463_v0_m_0</ModelName>
<CreateTime>2019-09-26 12:38:12</CreateTime>
<ExperimentId>4294</ExperimentId>
<UpdateTime>2019-09-26 12:38:12</UpdateTime>
<Project>pai_emr</Project>
<ProjectId>10009</ProjectId>
</Model>
</Models>
</Experiment>
</Experiments>
<RequestId>0a94818615695003656434743d0059</RequestId>
<ErrMsg>Successful</ErrMsg>
<ErrCode>success</ErrCode>
</ListPMMLModelsResponse>{
  "ListPMMLModelsResponse": {
    "Experiments": {
      "Experiment": {
        "ExpId": "4294",
        "Models": {
```

```

"Model": [
  {
    "Name": "Logistic regression for binary classification-1-Model",
    "Owner": "11769368777159105",
    "Description": "Logistic regression for binary classification-1-Model",
    "ModelId": "6119",
    "ModelName": "xlab_m_logisticregressi_51466_v0",
    "CreateTime": "2019-09-23 17:06:42",
    "ExperimentId": "4294",
    "UpdateTime": "2019-09-26 19:33:03",
    "Project": "pai_emr",
    "ProjectId": "10009"
  },
  {
    "Name": "Random forest1-AUC-0",
    "Owner": "11769368777159105",
    "Description": "Random forest1-AUC-0",
    "ModelId": "6200",
    "ModelName": "xlab_m_random_forests_1_51463_v0_m_0",
    "CreateTime": "2019-09-26 12:38:12",
    "ExperimentId": "4294",
    "UpdateTime": "2019-09-26 12:38:12",
    "Project": "pai_emr",
    "ProjectId": "10009"
  }
]
}
},
"RequestId": "0a94818615695003656434743d0059",
"ErrMsg": "Successful",
"ErrCode": "success"
}
}

```

4.5.2. Query detailed information about a PMML model

Operation name

DescribePMMLMode

Description

You can call this operation to query detailed information about a model.

Request parameters

Parameter	Type	Required	Example	Description
ModelId	Integer	Yes	6200	The ID of the model.

Parameter	Type	Required	Example	Description
Action	String	Yes	DescribePMMLMode	The operation that you want to perform. Set the value to DescribePMMLMode .

Response parameters

Parameter	Type	Description
Models	List<ModelInfo >	An array of model properties returned.
Model	ModelInfo	Detailed information about the model.
Name	String	The display name of the model.
Owner	String	The ID of the model owner.
Description	String	The description of the model.
ModelId	String	The ID of the model.
ModelName	String	The name of the underlying model generated by the algorithm.
CreateTime	Date	The time when the model was generated.
ExperimentId	String	The ID of the experiment from which the model is generated.
UpdateTime	Date	The time when the model was updated.
Project	String	The name of the project to which the model belongs.
ProjectId	String	The ID of the project to which the model belongs.

Sample requests

<http://pop.pai.idst.inter.env8d.com/?Action=DescribePMMLMode&ModelId=6200&<Common request parameters>>

Sample responses

```
<DescribePMMLModeResponse>
<Data>
<ModelInfo>
<Name>Logistic regression for binary classification-1-Model</Name>
<Owner>11769368777159105</Owner>
<ModelId>6119</ModelId>
<Description>Logistic regression for binary classification-1-Model</Description>
<ModelName>xlab_m_logisticregressi_51466_v0</ModelName>
<CreateTime>2019-09-23 17:06:42</CreateTime>
<UpdateTime>2019-09-26 19:33:03</UpdateTime>
<ExperimentId>4294</ExperimentId>
<Project>pai_emr</Project>
<ProjectId>10009</ProjectId>
</ModelInfo>
</Data>
<RequestId>0a94415315694992686868173d0065</RequestId>
<ErrMsg>Successful</ErrMsg>
<ErrCode>success</ErrCode>
</DescribePMMLModeResponse>{
  "DescribePMMLModeResponse": {
    "Data": {
      "ModelInfo": {
        "Name": "Logistic regression for binary classification-1-Model",
        "Owner": "11769368777159105",
        "ModelId": "6119",
        "Description": "Logistic regression for binary classification-1-Model",
        "ModelName": "xlab_m_logisticregressi_51466_v0",
        "CreateTime": "2019-09-23 17:06:42",
        "UpdateTime": "2019-09-26 19:33:03",
        "ExperimentId": "4294",
        "Project": "pai_emr",
        "ProjectId": "10009"
      }
    },
    "RequestId": "0a94415315694992686868173d0065",
    "ErrMsg": "Successful",
    "ErrCode": "success"
  }
}
```

4.5.3. Download PMML models

4.5.3.1. Generate a download URL for a model

Operation name

GeneratePMMLModelUrl

Description

You can call this operation to generate a download URL for a PMML model. The API call is executed asynchronously. After you call this operation, the system creates a download URL generation task in the background, and returns the task ID to you. You can then use the task ID to query the status of the task. If the task is successfully executed, the URL (OSS endpoint) of the model is returned.

Request parameters

Parameter	Type	Required	Example	Description
ModelId	Integer	Yes	6200	The ID of the model.
Action	String	Yes	GeneratePMMLModelUrl	The operation that you want to perform. Set the value to GeneratePMMLModelUrl.

Response parameters

Parameter	Type	Description
Data	Data	The returned data.
JobId	String	The ID of the download URL generation task.

Sample requests

```
http://pop.pai.idst.inter.env8d.com/?Action=GeneratePMMLModelUrl&ModelId=6200<Common request parameters>
```

Sample responses

```
<GeneratePMMLModelUrlResponse>
<Data>
<JobId>
asynUploadModel2Oss_395fd769-1568-4015-8aa8-a56b2e0da11c
</JobId>
</Data>
<RequestId>0a94818615695013054356697d0059</RequestId>
<ErrMsg>Successful</ErrMsg>
<ErrCode>success</ErrCode>
</GeneratePMMLModelUrlResponse>{
  "GeneratePMMLModelUrlResponse": {
    "Data": {
      "JobId": "
asynUploadModel2Oss_395fd769-1568-4015-8aa8-a56b2e0da11c
"
    },
    "RequestId": "0a94818615695013054356697d0059",
    "ErrMsg": "Successful",
    "ErrCode": "success"
  }
}
```

4.5.3.2. Query a model URL generation task

Operation name

QueryAsynJobStatus

Description

You can call this operation to query the status of a URL generation task. If the task is successfully executed, the URL (OSS endpoint) of the model is returned.

Request parameters

Parameter	Type	Required	Example	Description
JobId	String	Yes	asynUploadModel2Oss_395fd769-1568-4015-8aa8-a56b2e0da11c	The ID of the task to be queried. The task ID is returned after you call the GeneratePMMLModelUrl operation.
Action	String	Yes	QueryAsynJobStatus	The operation that you want to perform. Set the value to QueryAsynJobStatus.

Response parameters

Parameter	Type	Description
Data	Data	The returned data.
Status	String	The status of the task. Valid values: done, failed, and running.
Info	String	The URL of the model is returned if the task is successfully executed. An error message is returned if the system fails to execute the task.

Sample requests

```
http://pop.pai.idst.inter.env8d.com/?Action=QueryAsynJobStatus&JobId=asynUploadModel2Oss_395fd769-1568-4015-8aa8-a56b2e0da11c<Common request parameters>
```

Sample responses

```
<QueryAsynJobStatusResponse>
<Data>
<Status>done</Status>
<Info>
<![CDATA[
http://pai-global-oss.oss-cn-qingdao-env8d-d01-a.intra.env8d.com/xlab_m_random_forests_1_51463_v0_m_0-%E
9%9A%8F%E6%9C%BA%E6%A3%AE%E6%9E%971-AUC-0.pmml?Expires=1569501906&OSSAccessKeyId=yMPdrqaN
stzXKsNY&Signature=uRM1OFDZeDNzZo%2BjG87s09uUd6****
]]>
</Info>
</Data>
<RequestId>0a94818615695013421957579d0059</RequestId>
<ErrMsg>Successful</ErrMsg>
<ErrCode>success</ErrCode>
</QueryAsynJobStatusResponse>{
  "QueryAsynJobStatusResponse": {
    "Data": {
      "Status": "done",
      "Info": "
http://pai-global-oss.oss-cn-qingdao-env8d-d01-a.intra.env8d.com/xlab_m_random_forests_1_51463_v0_m_0-%E
9%9A%8F%E6%9C%BA%E6%A3%AE%E6%9E%971-AUC-0.pmml?Expires=1569501906&OSSAccessKeyId=yMPdrqaN
stzXKsNY&Signature=uRM1OFDZeDNzZo%2BjG87s09uUd6****
"
    },
    "RequestId": "0a94818615695013421957579d0059",
    "ErrMsg": "Successful",
    "ErrCode": "success"
  }
}
```

4.5.4. SDKs

Alibaba Cloud SDKs

```
https://developer.aliyun.com/sdk?spm=5176.10695662.1kquk9v2l.2.1e734735x9Gptc&aly_as=m3lFQpXP<dependen
cy>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<version>${version}</version></dependency>
```

SDK use case

Example: call an RPC API operation.

```
import com.aliyuncs.CommonRequest;
import com.aliyuncs.CommonResponse;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.profile.DefaultProfile;
public class Sample {
    public static void main(String[] args) {
        // Create a default ACS client and initialize it.
        DefaultProfile profile = DefaultProfile.getProfile(
            "<your-region-id>", // The region ID.
            "<your-access-key-id>", // The AccessKey ID.
            "<your-access-key-secret>"); // The AccessKey secret.
        IAcsClient client = new DefaultAcsClient(profile);
        // Create an API request and configure parameters.
        CommonRequest request = new CommonRequest();
        request.setDomain("ecs.aliyuncs.com");
        request.setVersion("2014-05-26");
        request.setAction("DescribeInstanceStatus");
        request.putQueryParameter("PageNumber", "1");
        request.putQueryParameter("PageSize", "30");
        try {
            CommonResponse response = client.getCommonResponse(request);
            System.out.println(response.getData());
        } catch (ServerException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (ClientException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

Example: call an RESTful API operation.

```
import com.aliyuncs.CommonRequest;
import com.aliyuncs.CommonResponse;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.profile.DefaultProfile;
public class Sample {
    public static void main(String[] args) {
        // Create a default ACS client and initialize it.
        DefaultProfile profile = DefaultProfile.getProfile(
            "<your-region-id>", // The region ID.
            "<your-access-key-id>", // The AccessKey ID.
            "<your-access-key-secret>");// The AccessKey secret.
        IAcsClient client = new DefaultAcsClient(profile);
        // Create an API request and configure parameters.
        CommonRequest request = new CommonRequest();
        request.setDomain("cs.aliyuncs.com");
        request.setVersion("2015-12-15");
        request.setUriPattern("/clusters");
        try {
            CommonResponse response = client.getCommonResponse(request);
            System.out.println(response.getData());
        } catch (ServerException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (ClientException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

SDK parameters

The domain must be in the pop.\${pai_domain} format. Set the pai_domain parameter to the domain shown in the Machine Learning Platform for AI.

The version must be 2019-09-25.

The Action string specifies an API operation.

The QueryParameter string specifies API request parameters, excluding the action parameter.

Example:

```
CommonRequest request = new CommonRequest();
request.setDomain("pop.${pai_domain}");
request.setVersion("2019-09-25");
request.setAction("DescribePMMLMode");
request.putQueryParameter("ModelId", "****");
```

4.6. Data labeling

4.6.1. Register a dataset

Machine Learning Platform for AI allows you to register a dataset by creating a dataset or importing a dataset file. It also allows you to use **manifest** files to manage all registered datasets.

Register a dataset by creating a dataset

If your source data such as image, text, video, and audio files is stored in Object Storage Service (OSS), you can **create a dataset** in the console. The system scans all files of the same type in the specified directory and generates a manifest file in the specified OSS path.

1. Go to the **Register Dataset** page.
 - i. In the left-side navigation pane of the Machine Learning Platform for AI console, choose **Data Preprocessing > Dataset Manager**.
 - ii. On the **Dataset Manager** page, click **Register Dataset**.
2. On the **Register Dataset** page, set the following parameters.

Parameter	Description
Dataset Name	The name must be 1 to 24 characters in length, and can contain underscores (_) and hyphens (-). It must start with a letter or a digit.
Method	Set the Method parameter to New Dataset .
Storage Type	Only OSS is supported.
Path	Set the Path parameter to the OSS path where your source data is stored.
Data Type	Only Image is supported.
Tags	You can attach a maximum of 10 tags to each dataset. Each tag can contain underscores (_) and hyphens (-). It must start with a letter or a digit.

3. Click **Submit**. Then, a manifest file is generated. The following code provides an example on the manifest file.

```

{"data":{"picUrl":"oss://****/pics/fruit/apple-1.jpg"}}
{"data":{"picUrl":"oss://****/pics/fruit/apple-10.jpg"}}
{"data":{"picUrl":"oss://****/pics/fruit/apple-11.jpg"}}
...
    
```

After you register a dataset, you can view the dataset on the **Dataset Manager** page.

Register a dataset by importing a dataset file

If you have created an on-premises **CSV file** or **manifest file**, you can register a dataset by **importing the dataset file**. If you import a CSV file, the system automatically converts it to a **manifest** file.

1. Go to the **Register Dataset** page.
 - i. In the left-side navigation pane of the Machine Learning Platform for AI console, choose **Data Preprocessing > Dataset Manager**.
 - ii. On the **Dataset Manager** page, click **Register Dataset**.
2. On the **Register Dataset** page, set the parameters.

Parameter	Description
Dataset Name	The name must be 1 to 24 characters in length, and can contain underscores (_) and hyphens (-). It must start with a letter or a digit.

Parameter	Description
Method	Set the Method parameter to Import Dataset .
Storage Type	Only OSS is supported.
Path	Select an OSS path.
Data Type	Drag an on-premises CSV file or manifest file to the upload area on the right side of the Data Type parameter. Note If the imported file is used in a labeling job, the names of the fields in the file must comply with the data structure of the template that is used to create the labeling job. For more information, see Data labeling templates .
Tags	You can attach a maximum of 10 tags to each dataset. Each tag must be 1 to 10 characters in length, and can contain underscores (_) and hyphens (-). It must start with a letter or a digit.

3. Click **Submit**. After you register a dataset, you can view the dataset on the **Dataset Manager** page.

4.6.2. Data labeling templates

Machine Learning Platform for AI provides the following templates: object detection, semantic segmentation, comprehensive image labeling, Optical Character Recognition (OCR), single-label image classification, and multi-label image classification. When you create a labeling job, you can select a template based on your requirements.

Object detection

Object detection is used to locate a specific object in an image. The rectangle selection tool is commonly used.

- Scenarios

Object detection applies to scenarios such as vehicle detection, passenger detection, and image search.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the `picUrl` field.

```
{"data":{"picUrl":"oss://****/pics/fruit/apple-1.jpg"}}  
...
```

- Output data

Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example on the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/pics/fruit/apple-1.jpg"
  },
  "label-****(Labeling job ID)": {
    "results": [{
      "data": [{
        "id": "Znyumd-****",
        "type": "image/rectangleLabel",
        "value": {
          "rotation": 0,
          "x": 40.68320610687023,
          "width": 327.52035623409665,
          "y": 5.762467474590647,
          "height": 296.68117192104745
        }
      },
      "labelColor": "#72bf7d",
      "labels": ["apple"]
    }],
    "id": "44****",
    "type": "image"
  }
}
```

Semantic segmentation

Semantic segmentation is used to recognize an object in an image and retrieve the coordinates of the object by scanning all pixels of the object. The commonly used tools are the polygon selection tool, brush tool, and superpixel tool.

- Scenarios

Semantic segmentation applies to scenarios such as autonomous driving, facial expression recognition, and apparel classification.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the `picUrl` field.

```
{"data":{"picUrl":"oss://****/pics/fruit/apple-1.jpg"}}
...
```

- Output data

Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example on the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/pics/fruit/apple-1.jpg"
  },
  "label-****(Labeling job ID)": {
    "results": [{
      "data": [{
        "id": "Znyumd-****",
        "type": "image/polygonLabel",
        "value": {
          "points": [
            [110, 46],
            [52, 196],
            [48, 168],
            [48, 145],
            [54, 120],
            [63, 93],
            [76, 74]
          ]
        }
      },
      {
        "labelColor": "#72bf7d",
        "labels": ["apple"]
      }
    ]},
    "id": "44****",
    "type": "image"
  }
}
```

Comprehensive image labeling

Comprehensive image labeling is used to match the content of the input images against a set of labels. This template allows you to use all image labeling tools as needed.

- Scenarios

Comprehensive image labeling applies to scenarios such as autonomous driving, content moderation, and content recognition.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the `picUrl` field.

```
{"data":{"picUrl":"oss://****/pics/fruit/apple-10.jpg"}}
```

- Output data

Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example on the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/pics/fruit/apple-10.jpg"
  },
  "label-****(Labeling job ID)": {
    "results": [{
      "data": [{
        "id": "Znyumd-****",
        "type": "image/rectangleLabel",
        "value": {
          "rotation": 0,
          "x": 40.68320610687023,
          "width": 327.52035623409665,
          "y": 5.762467474590647,
          "height": 296.68117192104745
        }
      }],
      "labelColor": "#72bf7d",
      "labels": ["Red apple"]
    }],
    "id": "44****",
    "type": "image"
  }
}
```

OCR

OCR is used to extract text from input images, and then classify the images based on the text.

- Scenarios

OCR applies to scenarios such as the recognition of identity cards, tickets, license plates, and bank cards.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the `picUrl` field.

```
{"data":{"picUrl":"oss://****/img/ocr_card/img0.jpeg"}}
```

- Output data

Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example on the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/img/ocr_card/img0.jpeg"
  },
  "label-****(Labeling job ID)": {
    "results": [
      {
        "data": [
          {
            "direction_of_picture": "downward",
            "type": "ocr/meta"
          },
          {
            "id": "Y4ZFoC-****",
            "direction_of_text": "downward",
            "text": "Alibaba Cloud Intelligence",
            "type": "ocr/polygonLabel",
            "value": {
              "points": [[325.08789110183716,397.47582054138184]]
            },
            "labelColor": "#67bd3a",
            "labels": "Enterprise"
          }
        ],
        "id": "24****",
        "type": "ocr"
      }
    ]
  }
}
```

Single-label image classification

Single-label image classification is used to find a label from a set of labels to match the content of an input image, and then attach the label to the image.

- Scenarios

Single-label image classification applies to scenarios such as photo classification, image recognition, and image search.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the `picUrl` field.

```
{"data":{"picUrl":"oss://****/img/ocr_card/img0.jpeg"}}
```

- Output data

Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example of the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/img/ocr_card/img0.jpeg"
  },
  "label-****(Labeling job ID)": {
    "results": [{
      "data": [{
        "data": "red",
        "id": "33****",
        "type": "survey/value"
      }],
      "id": "33****",
      "type": "survey"
    }]
  }
}
```

Multi-label image classification

Multi-label image classification is used to find multiple labels from a set of labels to match the content of an input image, and then attach the labels to the image.

- Scenarios

Multi-label image classification applies to scenarios such as content recommendation, advertising, and image search.

- Data structure

- Input data

Each row in the **manifest** file contains a topic. Each topic must contain the picUrl field.

```
{"data":{"picUrl":"oss://****/img/ocr_card/img0.jpeg"}}
```

- Each row in the **manifest** file contains a topic and the corresponding labeling result. The following code provides an example on the JSON string in each row:

```
{
  "data": {
    "picUrl": "oss://****/img/ocr_card/img0.jpeg"
  },
  "label-****(Labeling job ID)": {
    "results": [{
      "data": [{
        "data": ["red", "more", "green"],
        "id": "33****",
        "type": "survey/multivalue"
      }],
      "id": "33****",
      "type": "survey"
    }]
  }
}
```

4.6.3. Create a labeling job

This topic describes how to create a labeling job.

Prerequisites

A dataset is registered. For more information about how to register a dataset, see [Register a dataset](#).

Procedure

1. In the left-side navigation pane of the Machine Learning Platform for AI console, choose **Data Preprocessing > Smart Labeling**.
2. On the **Smart Labeling** page, click **Create Labeling Job**.
3. In the **Basic Information** step, set the parameters and click **Next**.

Parameter	Description
Task Name	The task name must be 1 to 24 characters in length, and can contain underscores (_) and hyphens (-). It must start with a letter or digit.
Description	The description must be 1 to 64 characters in length, and can contain underscores (_) and hyphens (-). It must start with a letter or digit.
Input Dataset	You can select one or more datasets to create a labeling job. The datasets must correspond to the topic of the labeling job. If no dataset is available, click Register Dataset next to the Input Dataset parameter to register a dataset.
Output Dataset Path	Select an Object Storage Service (OSS) path to store the labeling results. When you handle a labeling job, every time you click Generate Result Dataset , a result dataset is generated in the specified OSS path. The dataset contains the results of all topics that you have completed.

4. In the **Template** step, set the parameters and click **Next**.

Parameter	Description	
Template	The template to be used in the labeling job. The system supports the following templates: <ul style="list-style-type: none"> ◦ Target detection ◦ Semantic segmentation ◦ Image comprehensive labeling ◦ OCR ◦ Single-label image classification ◦ Multi-label image classification 	
Categories	The labels that are used to classify images. This parameter takes effect only when Single-label image classification or Multi-label classification is selected for the Template parameter.	
Label Image	Label Image Orientation	Specifies whether to annotate image orientation. This parameter takes effect only when you select OCR for the Template parameter.

Parameter	Description
Flip Text	Specifies whether to annotate text orientation. If the text in an image is placed in the same direction as the image, you can turn off this switch. This parameter takes effect only when you select OCR for the Template parameter.
Text Type	The labels that are used to classify the text. This parameter takes effect only when you select OCR for the Template parameter.
Add Custom Label	The custom labels that are used to classify the text. This parameter takes effect only when you select OCR for the Template parameter.
Labels	The labels that are used to classify images. Labels are displayed in different colors. This parameter takes effect only when you select Target detection , Semantic segmentation , or Image comprehensive labeling for the Template parameter.

5. In the **Labeling Policy** step, set the parameters and click **Submit**.

Parameter	Description
Dispatch Policy	The default dispatch policy is Number of topics collected by a worker each time and cannot be changed.
Topics per Collection	<p>The number of topics that are collected by each worker each time.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px;"> <p>? Note The value of the Topics per Collection parameter can be smaller than the total number of topics divided by the total number of workers. This allows workers with high efficiency to collect more topics and improves the overall efficiency of data labeling.</p> </div>
Add Worker	You can specify one or more workers. You can select both Alibaba Cloud accounts and RAM users.

4.6.4. Label images

This topic describes how to label images.

Prerequisites

A labeling job is created or assigned to you by the administrator. For more information, see [Create a labeling job](#).

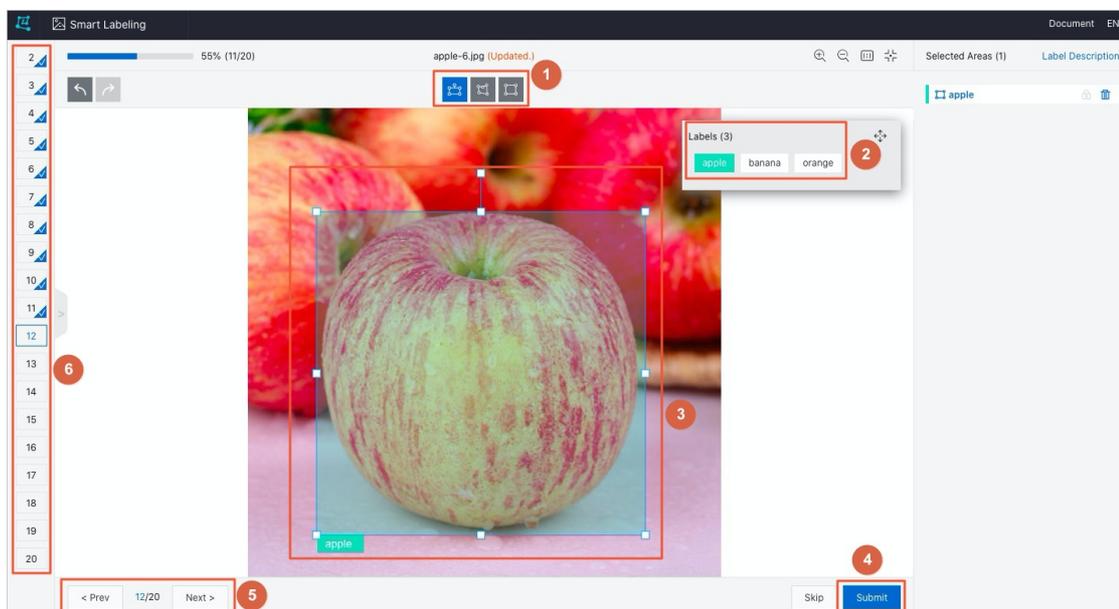
Procedure

1. Go to the Smart Labeling page.
 - i. In the left-side navigation pane of the Machine Learning Platform for AI console, choose **Data Preprocessing > Smart Labeling**.
 - ii. On the **Smart Labeling** page, click **My Labeling Jobs**.
 - iii. In the job list, find your labeling job and click **Start** in the **Actions** column.
2. Label images.
 - i. On the labeling page, click the  icon.

- ii. In the **Labels** section, select a label.

Note The system attaches the selected label to each image unless you select another label.

- iii. Use the selection tool to select an area in the image.



- iv. (Optional) If you do not need to label the image, click **Skip**.
- v. Click **Submit**.
- vi. You can use one of the following methods to browse and complete topics:
 - Click **Prev** or **Next** in the lower part of the Smart Labeling page.
 - Click the thumbnails of the topics in the left-side navigation pane.

4.7. EAS user guide

4.7.1. EAS overview

Elastic Algorithm Service (EAS) allows you to deploy machine learning models, such as PMML, PAI-OfflineModel, TensorFlow, and Caffe models, as services. You can also develop custom online services based on the API standards defined by EAS. You can simply use a JSON file to describe the service that you want to deploy, such as the path of the model, the region where the service is deployed, and the resources used by the service. After the JSON file is prepared, you can use the eascmd client or Alibaba Cloud SDK for Java to deploy it as an online service. The service can be accessed from the production environment and Internet. EAS supports various regions and different types of hardware resources (CPUs and GPUs).

4.7.2. Client

You can use the eascmd client to create, delete, view, and modify services. The eascmd client is stored in the Object Storage Service (OSS) bucket eas-utils. You can use wget or curl to download eascmd.

4.7.3. User authentication

Deployed model services use JSON Web Token (JWT) for user authentication. You can run the following command in eascmd to configure the AccessKey information.

```
./eascmd64 config -i <AccessKeyId> -k <AccessKeySecret> -e {{Domain name of EAS}}
//New users can use the admin account to pass authentication.
./eascmd64 config -i admin -k admin -e {{Domain name of EAS}}
```

4.7.4. Upload files

When you create a service, you must specify the HTTP addresses of the model and processor files. You can store the model and processor files on any HTTP servers.

4.7.5. Create a service

You can run the `create` command to create a service. When you create a service, you must specify the HTTP URLs of the model and processor files to be used by the service. You can upload the files to Object Storage Service (OSS) and specify their OSS paths.

```
eascmd create [service_desc_json]
```

The `service_desc_json` file contains service information and metadata configurations. The metadata configurations only contain cluster deployment information, such as the region where the cluster is deployed and the required resources. Example:

 **Note** When you run the `test` command to debug the service, the metadata configurations are ignored.

```
{
  "name": "mnist_saved_model_example",
  "generate_token": "true",
  "model_path": "http://eas-data.oss-cn-shanghai.aliyuncs.com/models%2Fmnist_saved_model.tar.gz",
  "processor": "tensorflow_cpu",
  "metadata": {
    "instance": 1,
    "cpu": 1,
  }
}
```

The parameters in the JSON file are described as follows:

Key	Value
name	The name of the service. Service names are unique inside each region. You can specify the region in the metadata configurations.
generate_token	Specifies whether to generate a token. If the value is set to true, you must add the token to the HTTP header of the request that you send to the service. If the value is set to false, the service is accessible to the public. No authentication is required.
token	Optional. The token used for authenticating requests sent to the service. If you do not specify a token, the <code>generate_token</code> parameter automatically generates a token.

Key	Value
model_path	The path of the model file. For more information, see the note under this table.
model_entry	Optional. The input file of the model. If no input file is specified, the file specified by the model_path parameter is used. You can specify arbitrary files in this parameter. The path of the primary file will be passed to the Load() function in the processor.
model_config	Optional. The configuration of the model. The configuration can be in any text format. The configuration is passed to the second parameter of the LoadWithConfig() function in the processor.
processor	Optional. The built-in processor used to make predictions. When this parameter is specified, the processor_path, processor_entry, processor_mainclass, and processor_type parameters are ignored.
processor_path	Optional. The path of the processor. Only custom processors support this parameter. For more information about processor packages, see the note under this table.
processor_entry	The primary file of the processor. It contains the implementations of the Load() and Process() functions. This parameter is required when you develop a custom processor using the C or C++ language.
processor_mainclass	The primary file of the processor. It defines the main class in Java. This parameter is required when you develop a custom processor using Java.
processor_type	Optional. The language used to develop the processor. Only custom processors support this parameter. Currently, only C++, Java, and Python are supported.
metadata	The metadata of the service.

Note model_path and processor_path specify the inputs of the model and processor in the format of HTTP URLs or OSS paths. When you use the test command to debug the service on your local machine, local paths are supported.

When HTTP URLs are used, the required files must be packaged in the tar.gz, tar.bz2, or ZIP format.

Metadata descriptions

Category	Parameter	Description
	workers	Optional. The number of threads used by each instance to process requests in parallel. The default is 5.

Category	Parameter	Description
Common options	instance	The number of instances launched by the service.
	CPU	The number of CPUs required by each instance.
	gpu	The number of GPUs required by each instance.
	resource	The name of the resource group. If your service uses CPU resources, ignore this parameter. Supported GPU resource groups are P4_4CORE and P4_8CORE.
Advanced parameter (use with caution)	rpc.batching	Optional. It specifies whether to enable batching at the server end for GPU optimization. The default is false.
	rpc.keepalive	Optional. The maximum amount of time that it takes to process a single request. When the time expires, the server end returns a 408 error and disconnects from the client. The default is 5000 milliseconds.
	rpc.io_threads	Optional. The number of threads used by each instance to handle network inputs and outputs. The default is 4.
	rpc.max_batch_size	Optional. The maximum size of each batch when batching is enabled. The default is 16.
	rpc.max_batch_timeout	Optional. The maximum timeout of each batch when batching is enabled. The default is 50 milliseconds.
	rpc.max_queue_size	Optional. The maximum size of the queue. The default is 64. When the queue is full, the server end returns a 450 error and closes the connection. The queue can prevent the server from being overloaded. It also can notify the client to send the request to other instances when the current instance is busy. For a queue that produces a long response time, you can set the size of the queue to a smaller value to prevent large amounts of pending requests from timing out.

Category	Parameter	Description
	rpc.worker_threads	Optional. The number of threads used by each instance to process requests in parallel. This parameter is the same as the workers parameter. The default is 5.

Example:

```
{
  ...
  "metadata": {
    "cpu": 4,
    "rpc.max_queue_size": 32,
    ...
  }
}
```

```
$ eascmd create pmml.json
[RequestId]: 1651567F-8F8D-4A2B-933D-F8D3E2DDEB2D
+-----+-----+
| Intranet Endpoint | http://pai-eas-vpc.cn-shanghai.aliyuncs.com/api/predict/savedmodel_exanple |
|   Token | YjQxZDYzZTBiZTZjMzQ5ZmE****   |
+-----+-----+
[OK] Creating api gateway
[OK] Building image [registry-vpc.cn-shanghai.aliyuncs.com/eas/savedmodel_exanple_cn-shanghai:v0.0.1-20190224001315]
[OK] Pushing image [registry-vpc.cn-shanghai.aliyuncs.com/eas/savedmodel_exanple_cn-shanghai:v0.0.1-20190224001315]
[OK] Waiting [Total: 1, Pending: 1, Running: 0]
[OK] Waiting [Total: 1, Pending: 1, Running: 0]
[OK] Service is running
```

4.7.6. Local debugging

Before you deploy the service to the cluster, you can use the local debugging feature to start a local service for debugging. This feature requires a Docker container and Internet access. You must install Docker on the machine where eascmd runs, and then run the following command to debug the service.

```
sudo eascmd test [service_desc_json]
```

Specify the JSON file used to deploy the service in this command. The following example shows how to run this command:

```
[xingke.***@mac-3:~/code/go/src/easgo]$ bin/eascmd test tf.json
[OK] Pulling image: registry.cn-shanghai.aliyuncs.com/eas/eas-worker-amd64:0.1.4
[OK] Pull image done
[OK] Creating container from: registry.cn-shanghai.aliyuncs.com/eas/eas-worker-amd64:0.1.4
[OK] Created container: e39176f85cf41a161bbb2896f76f1c0db0ad4f50e1d78a*****
[OK] Serving At: [http://localhost:6942/api/predict/savedmodel_exanple]
[2019-02-24 00:16:27] [172.17.0.2] Fetching model from [http://eas-data.oss-cn-shanghai.aliyuncs.com/models%2Ffl
ypig_scorecard.pmm]
[2019-02-24 00:16:27] [172.17.0.2] Fetching processor from [http://eas-data.oss-cn-shanghai.aliyuncs.com/eas-pmm
l-processor-0.1-jar-with-dependencies.jar]
[2019-02-24 00:16:28] [172.17.0.2] -----SERVICE LOG-----
...
[2019-02-24 00:16:30] [172.17.0.2] [INFO] Token: [WlzMFW5Jb8kckYj****]
...
[2019-02-24 00:16:30] [172.17.0.2] [INFO] Service start successfully
```

In this example, the command starts a local TensorFlow model service. The endpoint of the service is http://localhost:6942/api/predict/savedmodel_exanple.

4.7.7. Modify configurations

You can specify the `-D` parameter in the modify command to modify the metadata configurations, such as the instance, CPU, and memory configurations.

```
eascmd modify [service_name] -Dmetadata.[attr_name]=[attr_value]
```

For example, you can run the following command to set the number of instances to 10.

```
eascmd modify service_test -Dmetadata.instance=10
```

You can set multiple properties at a time. For example, you can set the number of instances to 10 and the memory size to 2,000 MB.

```
eascmd modify service_test -Dmetadata.instance=10 -Dmetadata.memory=2000
```

4.7.8. Modify a service

You can run the modify command to modify a deployed service.

```
eascmd modify [service_name] -s [service_desc_json]
```

 **Note** You cannot use this command to modify the region where the service is deployed. Note: If you only want to modify the resources used by the service, specify the metadata configurations in the service description file.

4.7.9. Delete a service

You can run the delete command to delete a service.

```
eascmd delete [service_name]
```

When you delete a service, you must specify the service name and the region where the service is deployed.

Examples:

```
$ eascmd delete savedmodel_exanple
Are you sure to delete the service [savedmodel_exanple] in [cn-shanghai]? [Y/n]
[RequestId]: 1651567F-8F8D-4A2B-933D-F8D3E2DDEB2D
[OK] Service [savedmodel_exanple] in region [cn-shanghai] is terminating
[OK] Service is terminating
[OK] Service is terminating
[OK] Service was deleted successfully
```

4.7.10. Switch service version

You can run the desc command to view the version of the current service and the latest version, and run the version command to switch the service to a version earlier than the latest version.

```
eascmd version [service_name] [version_id]
```

4.7.11. View service list

You can run the list (ls) command to view all services deployed by the current user.

```
$ eascmd ls
[RequestId]: 83945D4EED3E-4D35-A989-831E6BBA39F
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
|  SERVICENAME  | REGION | INSTANCE | CREATETIME | UPDATETIME | STATUS | WEIGHT | SERVICE PATH |
+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
| mnist_saved_model_example | cn-shanghai | 1 | 2019-02-21 16:35:41 | 2019-02-21 16:35:41 | Running | 0 | /api/predict/mnist_saved_model_example |
```

4.7.12. View service information

You can run the desc command to view detailed information about a deployed service.

```
eascmd desc [service_name]
```

Examples:

```

$ eascmd desc mnist_saved_model_example
+-----+
|   Status | Running
|   ServiceName | mnist_saved_model_example
|   Region | cn-shanghai
|   CreateTime | 2019-02-21 16:35:41
|   UpdateTime | 2019-02-21 16:35:41
|   AccessToken |
|   PrivateToken | ZWNjMTNkNDEzMmExNjZkYTM4YWQ5YTY****
|   TotalInstance | 1
|   RunningInstance | 1
|   PendingInstance | 0
|   CPU | 1
|   GPU | 0
|   Memory | 1000M
|   Image | registry-vpc.cn-shanghai.aliyuncs.com/eas/mnist_saved_model_example_cn-shanghai:v0.0.1-20190221163541
|   Weight | 0
|   LatestVersion | 1
|   CurrentVersion | 1
|   Message | Service start successfully
|   APiGatewayUrl | 1c3b37ea83c047efa0dc6df0cacb70d3-cn-shanghai.alicloudapi.com/EAPI_1828488879222746_mnist_saved_model_example
|   APiGatewayAppKey | 25641710
|   APiGatewayAppSecret | 12562a7b8858bbba****
|   IntranetEndpoint | http://pai-eas-vpc.cn-shanghai.aliyuncs.com/api/predict/mnist_saved_model_example
|
|   ServiceConfig | {
|       | "generate_token": "false",
|       | "metadata": {
|       |   | "cpu": 1,
|       |   | "instance": 1,
|       |   | "memory": 1000,
|       |   | "region": "cn-shanghai"
|       |   | },
|       | "model_path":
|       | "http://eas-data.oss-cn-shanghai.aliyuncs.com/models%2Fmnist_saved_model.tar.gz",
|       | "name":
|       | "mnist_saved_model_example",
|       | "processor":
|       | "tensorflow_cpu"
|       | }
|

```

4.7.13. View service processes

You can run the `showworkers(w)` command to view the status of the service processes.

```
eascmd w [service_name]
```

Examples:

```
$ eascmd w mnist_saved_model_example
[RequestId]: 4E905404-E617-4BD8-85D6-EC5C6A0D0211
+-----+-----+-----+-----+-----+-----+-----+
| INNERIP | HOSTIP | STARTAT | RESTARTS | STATUS | READY | REASON |
+-----+-----+-----+-----+-----+-----+-----+
| 172.24.5.183 | 192.168.65.121 | 2019-02-21 16:35:58 | 0 | Running | [1/1] | |
+-----+-----+-----+-----+-----+-----+-----+
```

4.7.14. Call a prediction service

This topic describes how to call a prediction service.

To call a prediction service, you must use the HTTP URL that is generated when you create the prediction service. The HTTP URLs for on-premises testing services are generated in the same format as those for cluster-based prediction services, except that the HTTP URLs use different hosts. When you send a request to a service URL, you must add a token to the HTTP header of the request. The token is generated when you create an image. The following example shows how to call a Predictive Model Markup Language (PMML) prediction service from cURL:

```
$ curl http://{{EAS domain name in the environment}}/api/predict/pmml_example -H 'Authorization: NGE3MzM4YmRhODZjMTE2NmZjZjNINTJINDgyNmZ3YzdjMmlwOD****==' -d '{}'
```

```
["prediction_score":0.0902926730924553]
```

You can use the following methods to obtain the EAS domain name in the environment:

- Concatenate a domain name:
 - To call the API for service management, you can concatenate a domain name in the following format:

```
eas.{{global:region}}.{{global:internet-domain}}
```

- To call the API for service calls, you can concatenate a domain name in one of the following formats based on whether the prediction service is deployed across regions:
 - Secondary region: eas.pai.{{global:region}}.{{global:internet-domain}}
 - Primary region: eas.pai.{{global:internet-domain}}
- Use the EAS Meta Server to obtain the access information by performing the following steps:
 - i. Run the following command to obtain the string with Region as the key:

```
curl http://easmeta.pai.{{global:internet-domain}}/regions
```

- ii. Find the access information of the current region.

The input and output formats of a prediction service are determined by the processor of the service.

4.7.15. Use Java or C++ to develop a model service

4.7.15.1. What are processors

Processors are program packages that contain the prediction logic. User requests are processed by processors and then returned to clients. A processor contains the logic for loading models and making predictions upon user requests. Machine Learning Platform for AI supports general processors: PMML and TensorFlow. If you want to customize the prediction logic, you must follow the processor development standards to develop a processor.

You can develop a processor in C, C++, and Java languages without the need to use an SDK. You only need to define the relevant classes and functions to develop a processor. This makes it easy for you to debug the processor offline.

4.7.15.2. C/C++ processor

If you use a C or C++ processor, you must define functions `Load()` and `Process()`. Function `Load()` is used to load the model during the initialization process. Function `Process()` is used to process service calls and return the processing results to clients. The two functions are declared as follows:

```
void *initialize(const char *model_entry, const char *model_config, int *state)
```

Parameter	Type	Description
model_entry	Input parameter	Specifies a model file. This parameter corresponds to the model_entry parameter in the configuration file when you deploy a model service. You can specify a file name or directory, for example, randomforest.pmml or ./model.
model_config	Input parameter	Specifies the custom model configurations. This parameter corresponds to the model_config parameter in the configuration file when you deploy a model service.
state	Output parameter	Indicates whether the model is loaded. Value 0 indicates that the model has been loaded.
N/A	Returned value	The memory address of the user-defined model variable. The value can be any data type.

```
int process(void *model_buf, const void *input_data, int input_size, void **output_data, int *output_size)
```

Parameter	Type	Description
model_buf	Input parameter	Specifies the memory address of the model returned by function initialize().
input_data	Input parameter	Specifies the input data. String data and binary data are supported.
input_size	Output parameter	Specifies the size of the input data.
output_data	Output parameter	The data returned by the processor. You must allocate heap memory to the processor. The model will automatically release the memory.

Parameter	Type	Description
output_size	Output parameter	The size of the data returned by the processor.
N/A	Returned value	If value 0 or 200 is returned, the model service is successfully called. This parameter can return HTTP error codes to clients. If the processor cannot recognize the HTTP error code, it automatically converts it to HTTP error code 400.

Examples

In the following example, no model data is loaded. The prediction service directly returns the request to the client.

```
#include <stdio.h>
#include <string.h>
extern "C" {
    void *initialize(const char *model_entry, const char *model_config, int *state)
    {
        *state = 0;
        return NULL;
    }
    int process(void *model_buf, const void *input_data, int input_size,
               void **output_data, int *output_size)
    {
        if (inputSize == 0) {
            const char *errmsg = "input data should not be empty";
            *outputData = strdup(errmsg, strlen(errmsg));
            *outputSize = strlen(errmsg);
            return 400;
        }
        *outputData = strdup((char *)inputData, inputSize);
        *outputSize = inputSize;
        return 200;
    }
}
```

The processor does not load any model data. It directly returns the input data to the client. You can use Makefile to compile the processor into a .so file.

```
CC=g++
CCFLAGS=-I./ -D_GNU_SOURCE -Wall -g -fPIC
LDFLAGS=-shared -Wl,-rpath=. /
OBJS=processor.o
TARGET=libpredictor.so
all: $(TARGET)
$(TARGET): $(OBJS)
$(CC) -o $(TARGET) $(OBJS) $(LDFLAGS) -L./
%.o: %.cc
$(CC) $(CCFLAGS) -c $< -o $@
clean:
rm -f $(TARGET) $(OBJS)
```

If the processor is reliant on other .so files, package these files with the processor.so file, and then deploy the package.

4.7.15.3. Java processor

The Java processor also uses functions Load() and Process() besides constructors. You only need to define one class for the Java processor. The class is defined as follows:

```
package com.alibaba.eas;
import java.util.*;
public class TestProcessor {
    public TestProcessor(String modelEntry, String modelConfig) {
        /* Passes a model file name to the class and initializes the model. */
    }
    public void Load() {
        /* Loads the model information based on the specified model file name */
    }
    public String Process(String input) {
        /* Preprocesses the input data and outputs the processing results. Currently, only string type data can be input and output */
    }
    public static void main(String[] args) {
        /* The main function is optional. It can be used for debugging on your local host. */
    }
}
```

If an exception occurs, the system automatically captures the exception, and returns an error message to the client. An HTTP 400 status code is also returned to the client. You can also customize the logic for capturing exceptions and return an error message to your client as follows:

```
try{
} catch (com.alibaba.fastjson.JSONException e) {
    throw new RuntimeException("bad json format, " + e.getMessage());
}
```

4.8. AutoML (must be activated separately)

4.8.1. Automatic parameter tuning with AutoML

This topic describes the automatic parameter tuning feature of AutoML.

Parameter

1. [Log in to machine learning console](#). In the left-side navigation pane, click **Experiments**.
2. Click an experiment to go to the canvas of the experiment.

 **Note** This topic uses air quality prediction as an example.

3. In the upper-left corner of the canvas, choose **Auto ML > Auto Parameter Tuning**.
4. On the **Auto Parameter Tuning** page, select an algorithm for parameter tuning, and click **Next**.

 **Note** You can select only one algorithm to tune at a time.

5. In the **Configure Parameter Tuning** module, set the **Parameter Tuning Method** parameter and click **Next**.

Alibaba Cloud Machine Learning Platform for AI provides four **parameter tuning methods**. For more information, see [Parameter adjustment method](#).

6. In the **Configure Model Output** module, set the model output parameters and click **Next**.

Parameter	Description
Evaluation Criteria	You can select one evaluation standard from the following four dimensions: AUC , F1-score , PRECISION , and RECALL .
Saved Models	You can save up to five models. The system ranks models based on the Evaluation Criteria setting you select and save the top ranked models according to the number entered in the Saved Models field.
Pass Down Model	This switch is turned on by default. If the switch is turned off, the model generated by the default parameters of the current component are passed down to the node of the subsequent component. If the switch is turned on, the optimal model generated by automatic parameter tuning are passed down to the node of the subsequent component.

7. In the upper-left corner of the canvas, click **Run** to run the automatic parameter tuning algorithm, as shown in the following figure.

 **Note** After the preceding configuration is complete, the Auto ML switch of the related algorithm is turned on. You can turn the switch on or off as needed.

8. Right-click a model component and choose **Edit AutoML Parameters** from the shortcut menu to modify its AutoML configuration parameters.

Output model display

1. During parameter tuning, right-click the target model component and choose **Parameter Tuning Details** from the shortcut menu.
2. On the **AutoML-Parameter Tuning Details** page, click the **Metrics** tab to view the current tuning progress and the running status of each model.
3. You can sort candidate models according to indicators (**AUC**, **F1-score**, **Accuracy**, and **Recall Rate**).
4. In the **View Details** column, you can click **Log** or **Parameter** to view the logs and parameters of each candidate model.

Parameter tuning effect display

1. On the **AutoML-Parameter Tuning Details** page, you can click the **Charts** tab to view the **Model Evaluation and Comparison** and **Hyperparameter Iteration Result Comparison** charts.
2. You can view the growth trend of the evaluation indicators of updated parameters in the **Hyperparameter Iteration Result Comparison** chart.

Model storage

1. [Log in to machine learning console](#). In the left-side navigation pane, click **Models**.

2. Click **Experiment Model** to open the experiment model folder.
3. Click the corresponding experiment folder to view the model saved with Auto ML.
4. (Optional) You can apply a model to other experiments by dragging the model to the canvas of the target experiment.

4.8.2. Parameter tuning methods

AutoML supports four parameter tuning methods.

Evolutionary Optimizer

Principle:

1. Randomly selects A parameter candidate sets (where A indicates the **number of exploration samples**).
2. Takes the N parameter candidate sets with higher evaluation indicators as the parameter candidate sets of the next iteration.
3. Continues the exploration within R times (where R indicates the **convergence coefficient**) as the standard deviation range around these parameters to explore new parameter sets. The new parameter sets replace the last A-N parameter sets by the evaluation indicator in the previous round.
4. Iterates the exploration for M rounds (where M indicates the **number of explorations**) until the optimal parameter set is found, according to the preceding logic.

Based on the preceding principle, the final number of models is $A + (A - N) \times M$.

 **Note** The first value of N is $A/2 - 1$. During iteration, the default value is $A/2 - 1$ (rounded up).

Parameter	Description
Data Splitting Ratio	Splits input data sources into training and evaluation sets. 0.7 indicates that 70% of the data is used for model training and 30% for evaluation.
Exploration Samples	The number of parameter sets of each iteration. The higher the number, the greater the accuracy, the larger the calculation. This parameter must be set to a positive integer in the range of 5 to 30.
Explorations	The number of iterations. The higher the number of iterations, the greater the search accuracy, the larger the calculation. This parameter must be set to a positive integer in the range of 1 to 10.
Convergence Coefficient	Tunes the exploration ranges (R times the standard deviation range search). The smaller the range, the faster the convergence (however, optimal parameters may be missed). Valid values: 0.1 to 1 (one floating point after the decimal point).

 **Note** You must enter the tuning range for each parameter. If the current parameter range is not configured, the parameter range is set by default.

Random Search

Principle:

1. Randomly selects a value for each parameter within the parameter range.
2. Enters random values into a set of parameters for model training.
3. Performs M rounds (where M indicates the **number of iterations**) and then sorts the output models.

Parameter	Description
Data Splitting Ratio	Splits input data sources into training and evaluation sets. 0.7 indicates that 70% of the data is used for model training and 30% for evaluation.
Iterations	The number of searches in the configured range. Valid values: 2 to 50.

 **Note** You must enter the tuning range for each parameter. If the current parameter range is not configured, the parameter range is set by default.

Grid Search

Principle:

1. Splits the value range of each parameter into N segments (where N indicates the **number of split grids**).
2. Randomly takes a value from the N segments. Assuming that there are M parameters, N^M parameter sets can be combined.
3. According to the N^M parameter sets, N^M models are generated by training. The models are then sorted.

Parameter	Description
Data Splitting Ratio	Splits input data sources into training and evaluation sets. 0.7 indicates that 70% of the data is used for model training and 30% for evaluation.
Grids	The number of split grids. Valid values: 2 to 10.

 **Note** You must enter the tuning range for each parameter. If the current parameter range is not configured, the parameter range is set by default.

Custom Parameters

- You can enumerate parameter candidate sets. The system then helps score all the combinations of the candidate sets.
- You can define enumeration ranges and separate parameters with commas (,). If the ranges are not configured, the default ranges of parameters are tuned.

4.9. Terms and acronyms

4.9.1. Terms

This topic lists the basic terms used in machine learning.

experiment

A user-created data mining workflow.

project

The basic object in MaxCompute. A project is also known as a workspace. A project contains other objects, such as tables and instances.

component

The minimum operating unit that you can invoke and execute on Apsara Stack Machine Learning Platform for AI. You can use components to import and export data, process data, analyze data, train models, and make predictions.

4.9.2. Acronyms

This topic describes the acronyms used in the Machine Learning Platform for AI User Guide.

MaxCompute

MaxCompute (formerly known as ODPS) is a data processing platform developed by Alibaba Cloud for large-scale data warehousing. MaxCompute can store and compute structured data in batches to meet the requirements of most big data modeling and analysis scenarios.

MaxCompute source and target tables

Tables are data storage objects in MaxCompute. Similar to relational database tables, tables in MaxCompute have a two-dimensional logical structure. A source table is the input of an algorithm node, while a target table is the output of an algorithm node.

5.E-MapReduce (EMR)

5.1. What is E-MapReduce?

E-MapReduce (EMR) is an end-to-end big data processing and analytics service. It uses open source big data ecosystems, such as Hadoop, Spark, Kafka, and Hive, to manage clusters, jobs, and data.

EMR is built in a virtual environment, such as ECS or Docker, and developed based on open source components, such as Hadoop, Spark, Hive, and Presto. EMR allows you to use peripheral systems, such as Hive, HBase, and Presto, in the Hadoop and Spark ecosystems to analyze and process data. EMR can also read data from or write data to other storage systems and database systems, such as MySQL and Oracle.

5.2. Introduction

5.2.1. Instructions

Before you use the E-MapReduce service, familiarize yourself with information such as the software configuration of the EMR cluster and the procedure for deploying the service.

5.2.2. Introduction

5.2.2.1. Software configuration

You need to install an operating system and big data components on each ECS instance in an E-MapReduce cluster.

5.2.2.2. Software requirements

This topic describes the software requirements of EMR.

The following table describes the software requirements.

Software requirements

Software	Description
Operating system	CentOS 7 64-bit kernel-3.10.0-693.2.2.el7.x86_64

5.2.2.3. Supported components

This topic lists the components of EMR.

The following table lists the components of EMR and the supported versions.

Components

Component	Version
Hadoop	2.8.5
Hive	3.1.1
Tez	0.9.1

Component	Version
Spark	2.4.3
Oozie	5.1.0
Hue	4.4.0
Zeppelin	0.8.1
Sqoop	1.4.7
ZooKeeper	3.5.5
Kafka	1.1.1
HBase	1.4.9
Phoenix	4.14.1
Presto	0.22.1
Flink	1.7.2
Ranger	1.2.0
Flume	1.8.0

Components

Component	Version
Hadoop	2.8.5
HDFS	2.8.5-1.4.0
Hive	3.1.1-1.1.6
Spark	2.4.3-1.2.0
Knox	1.1.0-1.0.2
Zeppelin	0.8.1-1.0.1
Tez	0.9.1-1.1.0
Ganglia	3.7.2
Pig	0.14.0
Sqoop	1.4.7-1.0.0
OpenLDAP	2.4.44
Hue	4.4.0
Yarn	2.8.5-1.4.0

Component	Version
Oozie	5.1.0
HBase	1.4.9-1.0.0
ZooKeeper	3.5.5
Presto	0.221
Impala	2.12.2-1.0.0
Superset	0.28.1
Ranger	1.2.0-1.1.0
Flink	1.10-vvr-1.0.4-SNAPSHOT
Storm	1.2.2
Phoenix	4.14.1
TensorFlow	1.8.0
Oozie	5.1.0
Kafka	2.11-1.1.1-1.3.0
Kafka-Manager	1.3.3.16-1.1.0
Druid	0.14.2-1.0.0

5.2.2.4. Introduction to components

This topic introduces the big data components that can run on an EMR cluster.

Component	Description
Hadoop	<ul style="list-style-type: none"> • YARN: supports task scheduling and cluster resource management. • HDFS: a distributed file system.
Hive	A Hadoop-based offline data processing system that provides an SQL-like interface to query data. Hive uses tables to store and manage data.
Spark	An in-memory distributed computing framework that supports offline and real-time computing, SQL statements, and machine learning.
Oozie	A job scheduler that supports workflow orchestration by building a directed acyclic graph (DAG). It supports multiple types of jobs.

Component	Description
Hue	A visualized platform used to manage open source components, such as Hadoop, Hive, Oozie, and HBase.
Sqoop	A tool designed to transfer data between HDFS and relational databases.
ZooKeeper	An open source distributed application coordination service. It is an open source implementation of Chubby provided by Google and is an important component of Hadoop and HBase. ZooKeeper solves the consistency problem of distributed applications. Its services include configuration maintenance, domain name services, distributed synchronization, and group services.
Kafka	A distributed messaging system that features high throughput, scalability, reliability, and performance. It is used in real-time computing, log processing, and data aggregation.
HBase	An open source, distributed, and column-oriented database. It is a component of the Apache Hadoop project. Different from typical relational databases, HBase is designed to store unstructured data. HBase is column-oriented rather than row-oriented.
Phoenix	Provides SQL-like statements for you to analyze HBase data.
Presto	A distributed SQL query engine that retrieves large datasets from one or more data sources.

Component	Description
Hadoop	<ul style="list-style-type: none"> • YARN: supports task scheduling and cluster resource management. • HDFS: a distributed file system.
Hive	A Hadoop-based offline data processing system that provides an SQL-like interface to query data. Hive uses tables to store and manage data.
Spark	An in-memory distributed computing framework that supports offline and real-time computing, SQL statements, and machine learning.
Knox	Provides service-level access control. You can manage the permissions to access a specific service by user, user group, or IP address.
Zepplin	Supports big data analytics and visualization.
Tez	Processes data based on DAGs of complex tasks.

Component	Description
Oozie	A job scheduler that supports workflow orchestration by building a DAG. It supports multiple types of jobs.
Hue	A visualized platform used to manage open source components, such as Hadoop, Hive, Oozie, and HBase.
Sqoop	A tool designed to transfer data between HDFS and relational databases.
ZooKeeper	An open source distributed application coordination service. It is an open source implementation of Chubby provided by Google and is an important component of Hadoop and HBase. ZooKeeper solves the consistency problem of distributed applications. Its services include configuration maintenance, domain name services, distributed synchronization, and group services.
Kafka	A distributed messaging system that features high throughput, scalability, reliability, and performance. It is used in real-time computing, log processing, and data aggregation.
HBase	An open source, distributed, and column-oriented database. It is a component of the Apache Hadoop project. Different from typical relational databases, HBase is designed to store unstructured data. HBase is column-oriented rather than row-oriented.
Impala	Enables fast and real-time interactive SQL queries on petabytes of data.
Flink (VVR)	VVR provides an enterprise-edition state backend whose performance is three to five times that of open source Flink. You can use the VVR engine and the EMR data development feature to submit jobs in an EMR Hadoop cluster.
Storm	A distributed framework for real-time big data processing.
Phoenix	Provides SQL-like statements for you to analyze HBase data.
TensorFlow	When you use TensorFlow to perform high-performance computing, you can allocate CPU and GPU resources by using YARN.
Presto	A distributed SQL query engine that retrieves large datasets from one or more data sources.

5.3. Log on to the EMR console

This topic describes how to log on to the EMR console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Context

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, choose **Products > Big Data > E-MapReduce**.
5. Set **Organization** and **Region** and click **EMR** to go to the EMR console.

5.4. Cluster management

5.4.1. Cluster planning

5.4.1.1. Gateway clusters

When you create a gateway cluster, you must associate it with an existing EMR cluster. Gateway clusters can serve as a separate job submission node to facilitate operations on the associated cluster.

A gateway cluster is an independent cluster that consists of multiple nodes with the same configurations. Clients, such as Hadoop (HDFS and YARN), Hive, Spark, and Sqoop clients, are deployed on the cluster.

If no gateway cluster is created, jobs of a Hadoop cluster are submitted on the master node or a core node of the Hadoop cluster, which consumes the resources of this cluster. After a gateway cluster is created, you can use it to submit jobs of the cluster associated with this gateway cluster. This way, the jobs do not occupy the resources of the associated cluster, and the stability of the master and core nodes, especially the master node, in the associated cluster is improved.

Each gateway cluster can have an independent configuration environment. For example, you can create multiple gateway clusters for one cluster that is shared by multiple departments to meet their different business requirements.

5.4.2. Cluster configuration

5.4.2.1. Create a cluster

This topic describes how to create an EMR cluster.

Procedure

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, click **Cluster Wizard** in the upper-right corner.
3. Configure cluster information. To create a cluster, you must configure software, hardware, and basic parameters as guided by the wizard.

 **Notice** After a cluster is created, you cannot modify its parameters except for the cluster name. Make sure that all parameters are correctly configured when you create a cluster.

i. Configure software parameters.

Parameter	Description
EMR Version	<p>The major version of EMR. It is a complete open source software environment. It can be regularly updated based on updates made to the internal component software.</p> <p>The major version of EMR is also updated with Hadoop-related software. Clusters of an earlier version cannot be automatically updated to a later version.</p>
Cluster Type	<p>The type of the cluster you want to create. EMR supports the following types of clusters:</p> <ul style="list-style-type: none"> ▪ Hadoop: Hadoop clusters provide multiple ecosystem components, such as Hadoop, Hive, Spark, Spark Streaming, Flink, Storm, Presto, Impala, Oozie, and Pig. Hadoop, Hive, and Spark are semi-hosted services and are used to store and compute large-scale distributed data offline. Spark Streaming, Flink, and Storm provide stream computing. Presto and Impala are used for interactive queries. ▪ Kafka: Kafka clusters serve as a semi-hosted, distributed messaging system with high throughput and scalability. Kafka clusters provide a comprehensive service monitoring system that maintains cluster stability. Kafka clusters are professional, reliable, and secure. You do not need to deploy or maintain these clusters. These clusters are used in scenarios such as log collection and monitoring data aggregation. They can also be used for offline data processing, stream computing, and real-time data analysis.
Required Services	<p>The default components required for a specific cluster type. After a cluster is created, you can add, start, or stop services on the Clusters and Services page of the cluster.</p>
Optional Services	<p>The other components that you can specify as required. By default, the relevant service processes for the components you specify are started.</p> <div style="background-color: #e1f5fe; padding: 10px; border: 1px solid #cfe2f3;"> <p> Note The more components you specify, the higher instance specifications a cluster needs to handle the components. You must select the instance type that matches the number of components you specified when you configure the hardware. Otherwise, the cluster may have insufficient resources to run the components.</p> </div>
Advanced Settings	<ul style="list-style-type: none"> ▪ Kerberos Mode: specifies whether to enable Kerberos authentication for clusters. This mode is disabled by default because it is not required by clusters created for common users. ▪ Custom Software Settings: customizes software settings. You can use a JSON file to customize the parameters of basic components required for a cluster, such as Hadoop, Spark, and Hive.

ii. Configure hardware parameters.

Section	Parameter	Description
---------	-----------	-------------

Section	Parameter	Description
Network Settings	Zone	<p>The zone where a cluster is created. Zones are different geographical areas located in the same region. They are interconnected by an internal network. The instance types and disks that are available for you to create a cluster depend on the selected zone.</p> <p>Networks of zones are isolated from each other. If your business requires a low-latency network, we recommend that you deploy instances in the same zone. However, if the zone has insufficient instances, a cluster may fail to be created.</p>
	Network Type	The network type of the cluster. The VPC network type is selected by default.
	VPC	The VPC where you want to deploy the cluster. Select a VPC in the same region as the zone.
	VSwitch	The vSwitch of the cluster. Select a vSwitch in the specified zone. If no vSwitch is available in the zone, create one.
	Security Group Name	<p>The security group to which you want to add your cluster. Only the security groups that you create in EMR are available in the drop-down list.</p> <p>To create a security group, you can directly enter a name in the Security Group Name field. The name must be 2 to 64 characters in length and can contain only letters, digits, underscores (_), and hyphens (-). It must start with a letter.</p>
High Availability	High Availability	<p>Specifies whether to enable the high availability mode. By default, this mode is disabled. In this case, the cluster has only one master node.</p> <p>For a Hadoop cluster, if the high availability mode is enabled, two master nodes are created in the cluster to ensure the availability of the ResourceManager and NameNode processes.</p>

Section	Parameter	Description
Instance	Configuration	<ul style="list-style-type: none"> ■ Node type: <ul style="list-style-type: none"> ■ Master Instance: runs control processes, such as ResourceManager and NameNode. ■ Core Instance: stores all the data of a cluster. You can add core nodes as needed after a cluster is created. ■ Instance type: <p>Select an instance type for each node type as needed.</p> ■ Data disk type: <p>Data disk types include standard SSD and ultra disk. Data disk types that are available for you to create the cluster vary based on the selected zone and instance type. By default, data disks are released after the cluster is released. For a compute node with local disks, the data disk type is selected by default and cannot be changed.</p> ■ Data disk size: <p>Valid values: 40 to 32768. Unit: GB. If you use a local disk, the size of the local disk is set by the system and cannot be modified.</p> ■ Node quantity: <p>You can set the number of nodes of each type as needed. A cluster with the high availability mode disabled must contain three nodes at least. If the high availability mode is enabled, another master node must be added to the cluster.</p>

iii. Configure basic parameters.

Section	Parameter	Description
Basic Information	Cluster Name	The name of the cluster. The name must be 1 to 64 characters in length and can contain only letters, digits, hyphens (-), and underscores (_).
	Assign Public Network IP	By default, this feature is disabled.
	Password	<p>The password used to log on to the master node. The password must be 8 to 30 characters in length and contain letters, digits, and special characters.</p> <p>The following special characters are supported:</p> <p>! @ # \$ % ^ & *</p>
Advanced Settings	Bootstrap Actions	Optional. You can configure bootstrap actions to run custom scripts before a cluster starts Hadoop.

5.4.2.2. Create a gateway cluster

A gateway cluster is composed of Elastic Compute Service (ECS) instances that reside in the same internal network as the EMR cluster associated with the gateway cluster. You can use a gateway cluster to achieve load balancing and security isolation and to submit jobs to the EMR cluster.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Procedure

1. [Log on to the EMR console](#).
2. In the upper-right corner of the **Cluster Management** page, click **Create Gateway**.
3. On the **Create Gateway** page, configure parameters.

Parameter	Description
Cluster Name	The name of the gateway cluster. The name must be 1 to 64 characters in length and can contain only letters, digits, hyphens (-), and underscores (_).
Assign Public IP Address	Specifies whether the gateway cluster is assigned an elastic IP address.
Password	The password used to log on to the gateway cluster. The password must be 8 to 30 characters in length and must contain uppercase letters, lowercase letters, digits, and special characters. The following special characters are supported: ! @ # \$ % ^ & *
Billing Method	The billing method of the gateway cluster. The default value is Pay-As-You-Go . The system charges you for a cluster based on the time the cluster is actually used in hours.
Associated Cluster	The EMR cluster associated with the gateway cluster. The gateway cluster submits jobs to this EMR cluster. The gateway cluster is automatically configured with the same Hadoop environment as the EMR cluster.
Zone	The zone where the associated cluster resides.
Network Type	The network type of the associated cluster.
VPC	The VPC to which the associated cluster belongs.
VSwitch	The vSwitch you want the gateway cluster to use. Select the vSwitch that corresponds to the zone and VPC.
Security Group Name	The name of the security group to which the associated cluster belongs.

4. After you configure the parameters, click **Create**. The created gateway cluster appears in the cluster list. Its state changes from **Initializing** to **Idle** a few minutes later.

5.4.2.3. Cluster list and details

This topic describes the parameters in the cluster list, parameters on the cluster details page, and supported cluster operations.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Cluster list

The cluster list displays the basic information about all of your clusters and the operations supported by each cluster.

Parameter	Description
Cluster ID/Name	The ID and name of a cluster. Move your pointer over a cluster name and click the  icon to modify the cluster name.
Cluster Type	The type of the cluster. Hadoop and Kafka are supported.
Status	The status of the cluster. If a cluster experiences an exception, such as a creation failure, error information appears in this column. You can move the pointer over the Help icon next to the error information to view detailed error information.
Created At	The time the cluster was created.
Time Elapsed	The duration from the point of creation to the current time. After the cluster is released, the time is no longer measured.
Actions	The operations that can be performed on the cluster. The supported operations include: <ul style="list-style-type: none"> • Manage: Go to the Clusters and Services page. • Details: View detailed information about the cluster on the Cluster Overview page.

Cluster details

The Cluster Overview page displays detailed information about the cluster and consists of four sections: Cluster Info, Software Info, Network Info, and Instance Info.

- Cluster Info

Cluster Info			
Cluster Name: emrtest	I/O Optimization: Yes	Start Time: Nov 11 11:11:11	Bootstrap Actions/EMR Version: EMR-3.22.1
Cluster ID: C-5	High Availability: No	Time Elapsed: 29 Days 23 Hours 11 Minutes 18 Seconds	ECS Role: AliyunEmrEcsDefaultRole
Region: cn-q	Security Mode: Standard		
Status:  idle			

Parameter	Description
Cluster Name	The name of the cluster.
Cluster ID	The ID of the cluster.

Parameter	Description
Region	The region where the cluster resides.
Status	The current status of the cluster.
I/O Optimization	Specifies whether I/O optimization is enabled.
High Availability	Specifies whether high availability is enabled.
Security Mode	The security mode. The software of the cluster starts in Kerberos security mode.
Start Time	The time the cluster was created.
Time Elapsed	The duration from the point of creation to the current time.
Bootstrap Actions/EMR Version	Information about custom scripts and software configuration.
ECS Role	The ECS application role. You can assign an application role to a cluster. This way, EMR applies for a temporary AccessKey pair when applications running on the compute nodes of that cluster access other Alibaba Cloud services, such as OSS. You do not need to manually enter an AccessKey pair. You can grant the application role access permissions on specific Alibaba Cloud services as required.

- Software Info

Software Info

EMR Version: EMR-3.22.1

Cluster Type: Hadoop

Software Info: [HDFS 2.8.5-1.4.0](#) [Yarn 2.8.5-1.4.0](#) [Hive 3.1.1-1.1.6](#) [Ganglia 3.7.2](#) [Spark 2.4.3-1.3.0](#) [Hue 4.4.0](#) [Tez 0.9.1-1.1.0](#) [Sqoop 1.4.7-1.0.0](#) [Pig 0.14.0](#) [Knox 1.1.0-1.0.2](#)

Parameter	Description
EMR Version	The major version of EMR.
Cluster Type	The type of the cluster.
Software Info	All the applications installed in the cluster and their versions.

- Network Info

Network Info

Region ID: cn-qing

Network Type: vpc

Parameter	Description
-----------	-------------

Parameter	Description
Region ID	The ID of the zone where the cluster resides, such as cn-hangzhou-b. The value is also the zone where the ECS instance resides.
Network Type	The network type of the cluster.

• Instance Info

ECS ID	Deployment Status	Public Network IP	Internal Network IP	Created At
i-9fx017	Normal		192.	Nov 10, 2020, 16:26:36

The following table describes the parameters of Master Instance Group.

Parameter	Description
Instances	The current number of nodes and the number of nodes that you have requested.
CPU	The number of vCPUs for a single node.
Memory	The memory capacity of a single node.
Data Disk Type	The data disk type and data disk capacity of a single instance.
ECS instances	The information about ECS instances in a master node group, including: <ul style="list-style-type: none"> ECS ID: the ID of a purchased ECS instance. Deployment Status: the deployment status of the ECS instance. Valid values: Initializing, Normal, and Scaling Out. Public Network IP: the public IP address of the ECS instance. Internal Network IP: the internal IP address of the ECS instance, which can be accessed by all nodes in the cluster. Created At: the creation time of the ECS instance.

5.4.2.4. View the running status of services

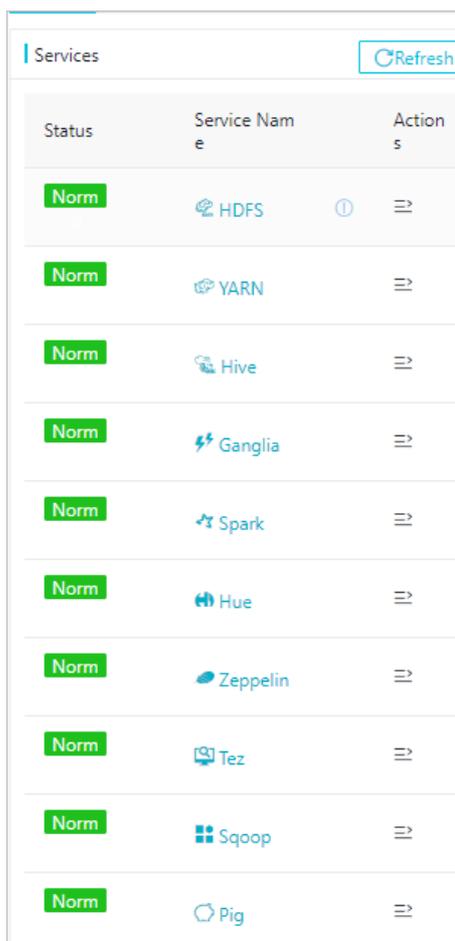
This topic describes how to view the running status of the services deployed in a cluster, such as HDFS and YARN, on the Clusters and Services page.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Procedure

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. On the **Clusters and Services** page, you can view the service list.



Status	Service Name	Actions
Norm	HDFS	ⓘ ≡
Norm	YARN	≡
Norm	Hive	≡
Norm	Ganglia	≡
Norm	Spark	≡
Norm	Hue	≡
Norm	Zeppelin	≡
Norm	Tez	≡
Norm	Sqoop	≡
Norm	Pig	≡

If a service, such as Storm, is not selected when you create the cluster, it is not displayed in the Services section.

You can click the name of a service to view its details, including the running status, deployment topology, configurations, and historical configuration modifications. A service may be in the **Normal** or **Error** state.

5.4.2.5. Configure component parameters

EMR allows you to modify and add parameters for components such as HDFS, YARN, Spark, Kafka, and Druid.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Modify parameters

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. In the **Services** section, click the component for which you want to modify parameters. HDFS is used as an example.
4. On the HDFS page, click the **Configure** tab.
5. Modify parameters.
 - i. In the **Configuration Filter** section, enter the keyword of the parameter you want to modify in the search box and click the  icon.

- ii. Find the parameter you want to modify and change its value.
6. Click **Save** in the upper-right corner. Perform the operations described in [Make configurations take effect](#) to make your modification take effect.

Add parameters

1. [Log on to the E-MapReduce console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. In the **Services** section, click the component for which you want to add parameters. **HDFS** is used as an example.
4. On the HDFS page, click the **Configure** tab.
5. Add parameters.
 - i. In the **Service Configuration** section, click the tab to which you want to add parameters.
 - ii. Click **Custom Configuration** in the upper-right corner.

Parameter	Description
Key	The name of the parameter that you want to add.
Value	The value of the parameter.
Description	The description of the parameter.
Actions	You can click Delete in this column to remove the parameter.
Add	You can click Add to add more parameters.

- iii. After the configuration is complete, click **OK**.
6. Click **Save** in the upper-right corner. Perform the operations described in [Make configurations take effect](#) to make the added parameters take effect.

Make configurations take effect

- Client-side configurations
 - i. After you modify or add client-side configurations, click **Deploy Client Configuration** in the upper-right corner of the **Configure** tab.
 - ii. In the **Cluster Activities** dialog box, specify **Target Nodes**, **Actions on Failures**, and **Description**.
 - iii. Click **OK**.
You can click **History** in the upper-right corner to view the execution status and progress.
- Server-side configurations
 - i. After you modify or add server-side configurations, click **Actions** in the upper-right corner of the **HDFS** page and select an option to restart the related service.
 - ii. Click **Deploy Client Configuration** in the upper-right corner.
 - iii. In the **Cluster Activities** dialog box, specify **Target Nodes**, **Actions on Failures**, and **Description**.
 - iv. Click **OK**.
You can click **History** in the upper-right corner to view the execution status and progress.

5.4.3. Restart a service or component

After you modify the configuration items of a service, you must restart the service or its components to make the configurations take effect. This topic describes how to restart a service or component in the EMR console.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Restart a service

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. In the left-side navigation pane of the Cluster Overview page, choose **Cluster Service > HDFS**. The HDFS service is used as an example in this topic.
4. In the upper-right corner of the HDFS page, choose **Actions > Restart All Components**.
5. In the **Cluster Activities** dialog box, specify **Description** and click **OK**.

Restart a component

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. In the left-side navigation pane of the Cluster Overview page, choose **Cluster Service > HDFS**. **DataNode** of the HDFS service is used as an example in this topic.
4. In the upper-right corner of the HDFS page, choose **Actions > Restart DataNode**.
5. In the **Cluster Activities** dialog box, specify **Description** and click **OK**.

5.4.4. View information about instances

This topic describes how to view the information about the instances of a cluster, including configurations of disks, vCPUs, and memory.

Prerequisites

A cluster is created. For more information, see [Create a cluster](#).

Procedure

1. [Log on to the EMR console](#).
2. On the **Cluster Management** page, find your cluster and click its **cluster ID**.
3. In the left-side navigation pane, click **Instances**. On the **Instances** page, you can view the information about the instances of the cluster, including configurations of disks, vCPUs, and memory.

6.DataHub

6.1. What is DataHub?

DataHub collects, stores, and processes streaming data, allowing you to analyze streaming data and build applications based on the streaming data.

DataHub is a platform designed to process streaming data. You can publish and subscribe to streaming data in DataHub and distribute the data to other platforms. DataHub allows you to analyze streaming data and build applications based on the streaming data.

DataHub collects, stores, and processes streaming data from mobile devices, applications, website services, and sensors. You can use your own applications or Apsara Stack Realtime Compute to process streaming data in DataHub, such as real-time website access logs, application logs, and events. The processing results such as alerts and statistics presented in graphs and tables are updated in real time.

Based on the Apsara system of Alibaba Cloud, DataHub features high availability, low latency, high scalability, and high throughput. DataHub is seamlessly integrated with Realtime Compute, allowing you to use SQL to analyze streaming data.

DataHub can also distribute streaming data to Apsara Stack services such as MaxCompute and Object Storage Service (OSS).

DataHub supports the following features:

- **Data queue:** DataHub automatically generates a cursor for each record in a shard, which can be considered as a logical data queue. The cursor is a unique sequence of numbers. You can improve the performance of a topic by increasing the number of shards in the topic.
- **Offset-based data consumption:** DataHub saves consumption offsets for applications. You can resume data consumption from a saved consumption offset when your application fails.
- **Data synchronization:** Data in DataHub can be automatically synchronized to other Apsara Stack services, including MaxCompute, OSS, AnalyticDB, ApsaraDB RDS for MySQL, Tablestore, and Elasticsearch.
- **Scalable topics:** DataHub allows you to scale in or out topics by splitting or merging shards.

6.2. Usage notes

Before you use DataHub, get familiar with the limits on specific features.

The following table describes the limits of DataHub.

Limits

Item	Limit	Description
Active shards per topic	(0,256]	Each topic can contain up to 256 active shards.
Shards	(0,512]	You can create up to 512 shards in each topic.
HTTP request body size	Up to 4 MB	The size of the HTTP request body cannot exceed 4 MB.
String size	Up to 1 MB	The size of a string cannot exceed 1 MB.

Item	Limit	Description
Merge and split operations on new shards	5s	You cannot merge a shard with another shard or split the shard within the 5s after the shard is created.
Queries per second (QPS)	Up to 5,000	The write QPS limit for each shard is 5,000. Multiple queries in one batch are considered as one query.
Throughput	Up to 5 MB/s	Each shard provides a throughput of up to 5 MB/s.
Projects	Up to 100	You can create up to 100 projects with each account.
Topics per project	Up to 1,000	You can create up to 1,000 topics in each project. Contact the administrator if you need to create more topics.
Time-to-live (TTL) of records	[1,7]	The TTL of each record in a topic ranges from one to seven days.

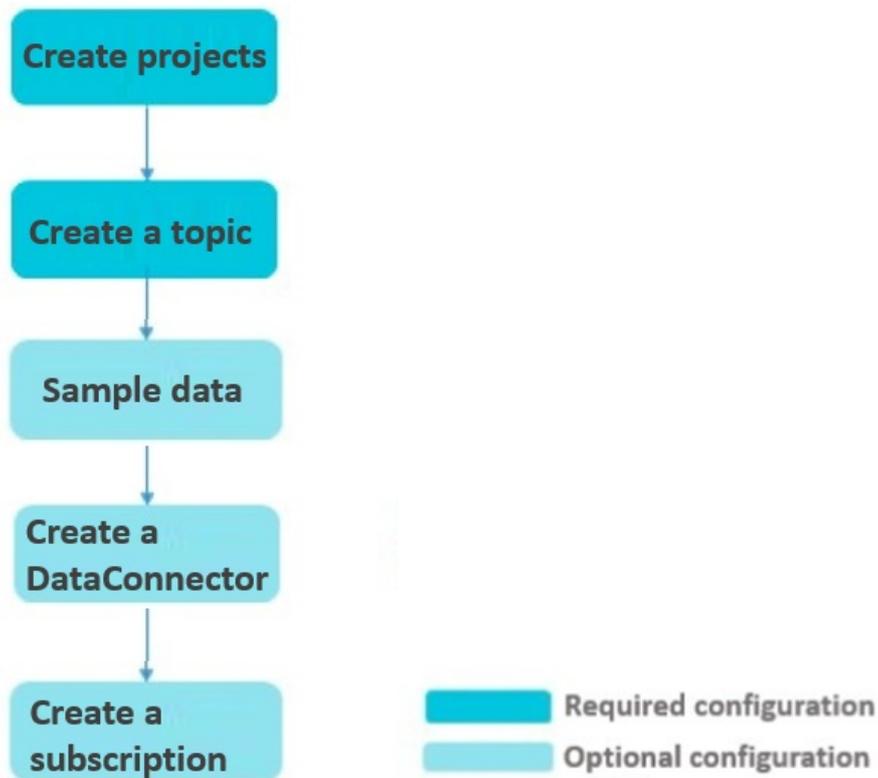
6.3. Quick Start

6.3.1. Overview

This topic describes the procedure of using DataHub.

Procedure shows the procedure of using DataHub.

Procedure



1. **Create projects.**

A project is an organizational unit in DataHub and contains one or more topics. When you use DataHub, you must create a project first.

2. **Create a topic.**

A topic is the smallest unit for data subscription and publication. You can use topics to distinguish different types of streaming data.

3. Optional. **Sample data.**

DataHub supports data sampling. You can sample data of a specific shard.

4. Optional. **Create a DataConnector.**

You can synchronize real-time data from DataHub to other data warehouses by using DataConnectors so that you can analyze and process historical data.

5. Optional. **Create a subscription.**

The subscription feature of DataHub supports saving consumption offsets on the server and allows applications to resume data consumption from saved consumption offsets. In addition, DataHub supports resetting offsets to ensure that data can be consumed at least once.

6.3.2. Log on to the DataHub console

This topic describes how to log on to the DataHub console. Google Chrome is used in this example.

Prerequisites

Before you log on to the DataHub console, make sure that the following requirements are met:

- You have obtained the URL of the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password from the operations administrator.

 **Note** If you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. To ensure the security of your account, the password must be 8 to 20 characters in length and must contain at least two types of the following characters:

- Uppercase or lowercase letters
- Digits
- Special characters: exclamation points (!), at signs (@), number signs (#), dollar signs (\$), and percent signs (%)

3. Click **Login** to go to the Apsara Uni-manager Management Console.
4. In the top navigation bar, choose **Products > Big Data > DataHub** to go to the DataHub console. The **Overview** page appears.

6.3.3. Create a project

A project is an organizational unit in DataHub and contains one or more topics. When you use DataHub, you must create a project first. This topic describes how to create a project in the DataHub console.

Prerequisites

An Apsara Stack tenant account is created.

Background information

- DataHub projects are independent from MaxCompute projects. Projects you created in MaxCompute cannot be used in DataHub.
- You can create up to 100 projects with each account.

Procedure

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, click **Create Project**. On the Create Project page, set parameters in the Region and Basic Settings sections and click **Submit**.

6.3.4. Create a topic

A topic is the smallest unit for data subscription and publication. You can use topics to distinguish different types of streaming data. This topic describes how to create a topic in the DataHub console.

Prerequisites

A project is created.

Background information

You can create up to 1,000 topics in each project. Contact the administrator if you need to create more topics.

Procedure

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click **View** in the Operate column.
3. On the project details page, click **Create Topic**.
4. In the Create Topic dialog box, set relevant parameters and click **Create**.

 **Note** DataHub allows you to directly create a topic or create a topic by importing a table schema from MaxCompute.

6.3.5. Sample data

DataHub supports data sampling. You can sample data of a specific shard. This topic describes how to sample data in the DataHub console.

Prerequisites

A project and a topic are created and data is written to the topic.

Background information

Before you sample data, you must specify the start time and the maximum number of records that you want to sample.

Procedure

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click **View** in the Operate column. On the project details page, find the target topic and click **View** in the Operate column.
3. On the Shard List tab of the topic details page, find the target shard and click **Sample** in the Operate column.
4. In the dialog box that appears, specify the start time and the maximum number of records that you want to sample and click **Sample**. DataHub samples the records that are written to the shard after the specified time and displays the sampled records in the table below **Sample**.

6.3.6. Create a DataConnector

You can synchronize real-time data from DataHub to other data warehouses by using DataConnectors so that you can analyze and process historical data. This topic describes how to create a DataConnector in the DataHub console.

Prerequisites

A project and a topic are created and data is written to the topic.

Background information

You can synchronize data from DataHub to MaxCompute, AnalyticDB, ApsaraDB RDS for MySQL, Tablestore, OSS, and Elasticsearch.

Procedure

1. Log on to the DataHub console.

2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click View in the Operate column. On the project details page, find the target topic and click **View** in the Operate column.
3. On the topic details page, click **Connector**. In the Create connector dialog box, select the data warehouse to which data is synchronized.
4. In the Create connector dialog box, set relevant parameters and click **Create**.

6.3.7. Create a subscription

The subscription feature of DataHub supports saving consumption offsets on the server and allows applications to resume data consumption from saved consumption offsets. In addition, DataHub supports resetting offsets to ensure that data can be consumed at least once. This topic describes how to create a subscription in the DataHub console.

Prerequisites

A project and a topic are created and data is written to the topic.

Procedure

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click View in the Operate column. On the project details page, find the target topic and click **View** in the Operate column.
3. On the topic details page, click **Subscription**. In the Create subscription dialog box, set relevant parameters and click **Create**.

6.4. Access Control

6.4.1. Overview

DataHub allows you to improve data security by granting different permissions to Apsara Stack tenant accounts and RAM user accounts.

DataHub uses Resource Access Management (RAM) for access control. Only users that have been granted the required permissions can access the resources in your department. By default, users do not have permission to access resources in your department. This topic describes how access control for DataHub is achieved by using RAM.

 **Note** An Apsara Stack tenant account is owned by a department and requires no authorization. A RAM user account must be granted permissions by the tenant account.

6.4.2. DataHub resources in RAM

The DataHub resources in RAM are project, topic, and subscription. Subscription is the action that you specify an application to read and process the records in DataHub. DataHub supports RAM authorization of project, topic, and subscription. RAM authorization is not supported at the shard level.

In RAM, each resource type has a general description and each specific object of the resource type has a description. For example, the description of a project that resides in a certain region is *acs:dhs:\$region:\$accountid:projects/\$projectName*. *\$region*, *\$accountid*, and *\$projectName* indicate the region that the project resides, the user ID, and the project name.

Resource description

Resource type	Description
SingleProject	acs:dhs:\$region:\$accountid:projects/\$projectName
AllProject	acs:dhs:\$region:\$accountid:projects/*
SingleTopic	acs:dhs:\$region:\$accountid:projects/\$projectName/topics/\$topicName
AllTopic	acs:dhs:\$region:\$accountid:projects/\$projectName/topics/*
SingleSubscription	acs:dhs:\$region:\$accountid:projects/\$projectName/topics/\$topicName/subscriptions/\$subid
AllSubscription	acs:dhs:\$region:\$accountid:projects/\$projectName/topics/\$topicName/subscriptions/*

6.4.3. API

DataHub provides application programming interfaces (APIs) for projects, topics, shards, subscriptions, and records. Before you can call the API operations, you must grant corresponding permissions to the RAM user by using RAM authorization policies.

The RAM authorization policy and resource type for each API operation is described as follows:

API operations for projects

API operations for projects

Operation name	RAM authorization policy	Resource type
CreateProject	dhs:CreateProject	AllProject
ListProject	dhs:ListProject	AllProject
DeleteProject	dhs>DeleteProject	SingleProject
GetProject	dhs:GetProject	SingleProject
UpdateProject	dhs:UpdateProject	SingleProject

API operations for topics

API operations for topics

Operation name	RAM authorization policy	Resource type
CreateTopic	dhs:CreateTopic	AllTopic
ListTopic	dhs:ListTopic	AllTopic
DeleteTopic	dhs>DeleteTopic	SingleTopic
GetTopic	dhs:GetTopic	SingleTopic
UpdateTopic	dhs:UpdateTopic	SingleTopic

API operations for subscriptions

API operations for subscriptions

Operation name	RAM authorization policy	Resource type
CreateSubscription	dhs:CreateSubscription	AllSubscription
ListSubscription	dhs:ListSubscription	AllSubscription
DeleteSubscription	dhs>DeleteSubscription	SingleSubscription
GetSubscription	dhs:GetSubscription	SingleSubscription
UpdateSubscription	dhs:UpdateSubscription	SingleSubscription
CommitOffset	dhs:CommitOffset	SingleSubscription
GetOffset	dhs:GetOffset	SingleSubscription

API operations for shards

API operations for shards

Operation name	RAM authorization policy	Resource type
ListShard	dhs:ListShard	SingleTopic
MergeShard	dhs:MergeShard	SingleTopic
SplitShard	dhs:SplitShard	SingleTopic

API operations for shards

API operations for shards

Operation name	RAM authorization policy	Resource type
PutRecords	dhs:PutRecords	SingleTopic
GetRecords	dhs:GetRecords	SingleTopic
GetCursor	dhs:GetRecords	SingleTopic

6.4.4. Conditions

This section describes conditions that can be applied to the RAM authorization policies for DataHub.

Conditions that can be applied to the RAM authorization policies for DataHub are as follows:

RAM authorization policy conditions for DataHub

Condition keyword	Description	Valid value
acs:SourceIp	The IP address range that can access the specified object.	Any valid IP address. Wildcard masks are supported.

Condition keyword	Description	Valid value
acs:SecureTransport	Indicates whether HTTPS is used to access the specified object.	true/false
acs:MFAPresent	Indicates whether the specified object can be accessed by multiple clients.	true/false
acs:CurrentTime	The time that the specified object can be accessed.	This keyword must be described in ISO 8601 format.

6.4.5. Sample RAM authorization policy content

6.4.5.1. AliyunDataHubFullAccess

This section describes how to set the AliyunDataHubFullAccess policy content.

The authorization policy content can be set as follows:

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": "dhs:*",
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

6.4.5.2. AliyunDataHubReadOnlyAccess

This section describes how to set the AliyunDataHubReadOnlyAccess policy content.

The authorization policy content can be set as follows:

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": ["dhs:List*", "dhs:Get*"],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

6.5. Data Acquisition

6.5.1. Overview

In addition to SDK and local file uploads, DataHub supports various data acquisition tools to help you quickly collect data to DataHub.

This section describes how to acquire data by using Fluentd, Logstash, and Oracle GoldenGate (OGG).

6.5.2. Fluentd

This topic describes how to install and use the DataHub plug-in for Fluentd.

Developed based on the open-source data collector Fluentd, the DataHub plug-in for Fluentd is easy to install and is used to write the collected data to DataHub.

Install the DataHub plug-in for Fluentd

- Install the plug-in by using RubyGems

```
gem install fluent-plugin-datahub
```

 **Notice** We recommend that you change the gem source to <https://ruby.taobao.org/>.

- Install the plug-in by using a local installation package

The agent must be installed in Linux. Before you install the agent, install Ruby. For users who have not installed Fluentd, a full installation package for installing both Fluentd and DataHub plug-in is provided. For users who have installed Fluentd, an installation package of the DataHub plug-in is provided.

- If you have not installed Fluentd, download the [full installation package](#) and run the following commands to install Fluentd with the DataHub plug-in:

 **Notice** Fluentd 0.12.23 is provided in the full installation package.

```
$ tar -xzf fluentd-with-datahub-0.12.23.tar.gz
$ cd fluentd-with-datahub
$ sudo sh install.sh
```

- If you have installed Fluentd, download the [installation package of the DataHub plug-in for Fluentd](#) and run the following command to install the plug-in.

```
$ sudo gem install --local fluent-plugin-datahub-0.0.2.gem
```

Use cases

Case 1: Collect CSV files

This example shows how to write the incremental content of a Comma-Separated Values (CSV) file to DataHub in quasi-real time by using the DataHub plug-in for Fluentd. The following CSV file is used in this example:

```
0,qe614c760fuk8judu01tn5x055rpt1,true,100.1,14321111111
1,znv1py74o8ynn87k66o32ao4x875wi,true,100.1,14321111111
2,7nm0mtpgo1q0ubuljjx9b000ybltl,true,100.1,14321111111
3,10t0n6pvonnan16279w848ukko5f6l,true,100.1,14321111111
4,0ub584kw88s6dczd0mta7itmta10jo,true,100.1,14321111111
5,1ltpf0jt7fhvf0oy4lo8m3z62c940,true,100.1,14321111111
6,zpqsfxqy9379lmcehd7q8kftntrozbt,true,100.1,14321111111
7,ce1ga9aln346xcj761c3iytshyzuxg,true,100.1,14321111111
8,k5j2id9a0ko90cykl40s6ojq6gruyi,true,100.1,14321111111
9,ns2zcx9bdip5y0aqd1tdicf7bkdmsm,true,100.1,14321111111
10,54rs9cm1xau2fk66pzyz62tf9tsse4,true,100.1,14321111111
```

Each line is a record to be written to DataHub. Columns are separated by commas (.). Save the CSV file as /temp/test.csv on the local computer. The following table shows the schema of the DataHub topic to which the CSV file is written.

DataHub topic schema

Field	Data type
id	BIGINT
name	STRING
gender	BOOLEAN
salary	DOUBLE
my_time	TIMESTAMP

After you edit the Fluentd configuration file based on the CSV file and topic schema, run the following command to start Fluentd to write the CSV file to DataHub:

```
`${FLUENTD_HOME}/fluentd-with-dataHub/bin/fluentd -c fluentd_test.conf
```

Use the following Fluentd configuration file in this example:

```
<source>
@type tail
path /xxx/yyy (Specify the path of the CSV file.)
tag test1
format csv
keys id,name,gender,salary,my_time
</source>
<match test1>
@type dataHub
access_id your_app_id
access_key your_app_key
endpoint http://ip:port
project_name test_project
topic_name fluentd_performance_test_1
column_names ["id", "name", "gender", "salary", "my_time"]
flush_interval 1s
buffer_chunk_limit 3m
buffer_queue_limit 128
dirty_data_continue true
dirty_data_file /xxx/yyy (Specify the path of the dirty record file.)
retry_times 3
put_data_batch_size 1000
</match>
```

Case 2: Collect Log4j logs

The following format of Log4j logs is used in this example:

```
11:48:43.439 [qtp1847995714-17] INFO AuditInterceptor - [c2un5sh7cu52ek6am1ui1m5h] end /web/v1/project/tefe4mfurtix9kwwyrvfqd0m/node/0m0169kapshvgc3ujskwkk8g/health GET, 4061 ms
```

Use the following Fluentd configuration file in this example:

```
<source>
@type tail
path bayes.log
tag test
format /(? <request_time>\d\d:\d\d:\d\d.\d+)\s+[(? <thread_id>[w-]+)]\s+(? <log_level>w+)\s+(? <class>w+)\s+[\s+](? <request_id>w+)\s+(? <detail>.+)/
</source>
<match test>
@type dataHub
access_id your_access_id
access_key your_access_key
endpoint http://ip:port
project_name test_project
topic_name dataHub_fluentd_out_1
column_names ["thread_id", "log_level", "class"]
</match>
```

Parameter description

Input configuration

Parameter	Description
tag test1	The tag, which is mapped to the destination information by using the specified regular expression.
format csv	The format of the file from which data is collected.
keys id,name,gender,salary,my_time	The columns to be collected from the CSV file. The column names must be the same as those in the schema of the destination DataHub topic.

Output configuration

Parameter	Description
shard_id 0	The ID of the shard to which all records are written. By default, all records are written to the shard by polling. The default ID is 0.
shard_keys ["id"]	The column used as the shard key. Hashed shard key values are used as indexes for writing data.
flush_interval 1	The interval between data flushes. The default value is 60s.
buffer_chunk_limit 3m	The maximum size of a chunk. Unit: k or m, which indicates KB or MB. We recommend you set the maximum size to 3 MB.
buffer_queue_limit 128	The maximum length of the chunk queue. Both the buffer_chunk_limit and buffer_queue_limit parameters determine the size of the buffer. The default value is 128 MB.
put_data_batch_size 1000	The number of records to be written to DataHub at a time. In this example, 1,000 records are written to DataHub each time.
retry_times 3	The number of retries for writing data to DataHub. Default value: 3.
retry_interval 3	The retry interval at which data is written. Unit: seconds. Default value: 3.
dirty_data_continue true	Specifies whether to ignore dirty records. The value true indicates that the plug-in retries the operation for a specified number of times before it writes the dirty records to the dirty record file.
dirty_data_file /xxx/yyy	The directory where the dirty record file is stored.
column_names ["id"]	The name of the columns to be written to DataHub.

6.5.3. Logstash

This topic describes how to install and use Logstash to import data to DataHub and export data from DataHub.

Logstash is a distributed log collection framework. It is often used with Elasticsearch and Kibana, known as the ELK Stack, for log data analysis. To support a wider variety of data inputs, DataHub offers Output and Input plug-ins for data transfer with Logstash. By using Logstash, you can access more than 30 types of data sources in the Logstash open source community, such as files, Syslog logs, Redis logs, Log4j logs, Apache logs, and NGINX logs. Logstash also supports filter plug-ins for customizing the fields to be transferred. This topic demonstrates how to use Logstash with DataHub.

Install Logstash and DataHub plug-ins

Java Runtime Environment (JRE) 7 or later is required to run Logstash. If the JRE version does not meet the requirement, several features of Logstash are unavailable. You can install Logstash and DataHub plug-ins with one click by downloading and decompressing the software package or install Logstash and DataHub plug-ins separately.

- Install Logstash and DataHub plug-ins with one click: Download the [software package](#).

Run the following commands to decompress the package and go to the software directory:

```
$ tar -xvf logstash-with-datahub-2.3.0.tar.gz
$ cd logstash-with-datahub-2.3.0
```

- Install Logstash and DataHub plug-ins separately
 - Install Logstash. For more information, see the [documentation on the official website of Logstash](#).
 - Install the [DataHub Output plug-in for Logstash](#). You can use this plug-in to import data to DataHub.
 - Install the [DataHub Input plug-in for Logstash](#). You can use this plug-in to export data from DataHub.

Use cases

Case 1: Collect Log4j logs

This example shows how to collect unstructured Log4j logs and derive a structure out of the logs by using Logstash. The following format of Log4j logs is used in this example:

```
20:04:30.359 [qtp1453606810-20] INFO AuditInterceptor - [13pn9kdr5tl84stzkmaa8vmg] end /web/v1/project/fhp4clxfbu0w3ym2n7ee6ynh/statistics? executionName=bayes_poc_test GET, 187 ms
```

In this example, you can derive a structure out of the logs and transfer the data to DataHub. The following table shows the schema of the DataHub topic to which the Log4j logs are written.

DataHub topic schema

Field	Data type
request_time	STRING
thread_id	STRING
log_level	STRING
class_name	STRING
request_id	STRING
detail	STRING

Use the following configuration of the Logstash task in this example:

```

input {
  file {
    path => "${APP_HOME}/log/bayes.log"
    start_position => "beginning"
  }
}
filter{
  grok {
    match => {
      "message" => "(? <request_time>\d\d:\d\d:\d\d\.\d+)\s+[[? <thread_id>[\w\.-]+]]\s+(? <log_level>\w+)\s+(? <class_name>\w+)\s+[\s+][[? <request_id>\w+]]\s+(? <detail>.+)"
    }
  }
}
output {
  datahub {
    access_id => "Your accessId"
    access_key => "Your accessKey"
    endpoint => "Endpoint"
    project_name => "project"
    topic_name => "topic"
    #shard_id => "0"
    #shard_keys => ["thread_id"]
    dirty_data_continue => true
    dirty_data_file => "/Users/ph0ly/trash/dirty.data"
    dirty_data_file_max_size => 1000
  }
}

```

Case 2: Collect CSV files

This example shows how to use Logstash to collect CSV files. The following CSV file is used in this example:

```

1111,1.23456789012E9,true,14321111111000000,string_dataxxx0,
2222,2.23456789012E9,false,14321111111000000,string_dataxxx1

```

The following table shows the schema of the DataHub topic to which the CSV file is written.

DataHub topic schema

Field	Data type
col1	BIGINT
col2	DOUBLE
col3	BOOLEAN
col4	TIMESTAMP
col5	STRING

Use the following configuration of the Logstash task in this example:

Run the following command to start Logstash:

```
$LOGSTASH_HOME/bin/logstash -f <The preceding configuration file> -b 256
```

Note -f is followed by the path of the configuration file. -b is followed by the number of records transferred to DataHub at a time. Default value: 125.

Parameters

The following table describes the parameters of the DataHub Output plug-in.

Parameters of the DataHub Output plug-in

Parameter	Description
access_id	Required. The AccessKey ID of your Apsara Stack tenant account.
access_key	Required. The AccessKey secret of your Apsara Stack tenant account.
endpoint	Required. The endpoint used to access DataHub.
project_name	Required. The name of the DataHub project.
topic_name	Required. The name of the DataHub topic.
retry_times	Optional. The maximum number of retries. The value -1 indicates unlimited retries. The value 0 indicates no retries. A value greater than 0 indicates the specified number of retries. Default value: -1.
retry_interval	Optional. The interval between retries. Unit: seconds. Default value: 5.
shard_keys	Optional. The key of the shard. The hash of the key value is used to map the ID of the shard to which the records are written. If the shard_keys and shard_id parameters are not specified, the system polls the shards to decide which shard the records are written to.
shard_id	Optional. The ID of the shard where records are written. If the shard_keys and shard_id parameters are not specified, the system polls the shards to decide which shard the records are written to.
dirty_data_continue	Optional. Specifies whether to ignore dirty records. The value true indicates that dirty records are to be ignored. Default value: false. If you set the value to true, you must specify the dirty_data_file parameter.
dirty_data_file	Optional. The name of the dirty record file. The dirty record file is divided into .part 1 and .part 2. The most recent records are stored in .part 2.
dirty_data_file_max_size	Optional. The maximum size of the dirty record file. This value is for reference only.

The following table describes the parameters of the DataHub Input plug-in.

Parameters of the DataHub Input plug-in

Parameter	Description
access_id	Required. The AccessKey ID of your Apsara Stack tenant account.

Parameter	Description
access_key	Required. The AccessKey secret of your Apsara Stack tenant account.
endpoint	Required. The endpoint used to access DataHub.
project_name	Required. The name of the DataHub project.
topic_name	Required. The name of the DataHub topic.
retry_times	Optional. The maximum number of retries. The value -1 indicates unlimited retries. The value 0 indicates no retries. A value greater than 0 indicates the specified number of retries. Default value: -1.
retry_interval	Optional. The interval between retries. Unit: seconds. Default value: 5.
shard_ids	Optional. The shards in which records are to be consumed. If this parameter is not specified, records in all the shards are consumed.
cursor	Optional. The sequence number of the record from which the consumption begins. The consumption starts from the earliest record by default.
pos_file	Required. The checkpoint file, which is used to reset the consumption offset.

6.5.4. Oracle GoldenGate

This topic describes how to install and use Oracle GoldenGate (OGG).

OGG is a tool for log-based structured data replication across heterogeneous environments. It is used for data backup between primary and secondary Oracle databases. It is also used to synchronize data from Oracle databases to other databases such as IBM Db2 and MySQL databases. OGG must be deployed in the source and destination databases. It is composed of the following components: Manager, Extract, data pump, Collector, and Replicat.

- Manager is the control process of OGG. A Manager process must be running on the source and destination databases. It is responsible for starting, stopping, and monitoring other processes.
- Extract is a process that captures data from the source database or transaction logs. You can configure the Extract process for initial data loads and incremental data synchronization. For initial data loads, Extract captures a set of data directly from their source objects. To keep source data synchronized to the destination database, Extract captures incremental DML and DDL operations after the initial data loading has taken place. This topic describes incremental data synchronization.
- A data pump is a secondary Extract group within the source OGG configuration. In a typical configuration with a data pump, the primary Extract group writes to a trail on the source database. The data pump reads the trail and sends the DML or DDL operations over the network to a remote trail on the destination database.
- Collector is a process on the destination database, which receives data from the source database and generates trail files.
- Replicat is a process that reads the trail on the destination database, reconstructs the DML or DDL operations, and then applies them to the destination database.

The DataHub agent for OGG offers the Replicat feature that applies the updated data to DataHub by analyzing the trail. The data in DataHub is processed in real time by using Realtime Compute and can be archived into MaxCompute.

The following example shows how to synchronize incremental data from an Oracle database to DataHub and process the data in DataHub.

Install OGG

Prerequisites:

- You have installed the Oracle database client.
- You have obtained the OGG installation package for the source database. We recommend that you use OGG V12.1.2.1.
- You have obtained the OGG Adapters installation package for the destination database. We recommend that you use OGG Application Adapters 12.1.2.1.
- You have installed Java 7.

Follow these steps to install OGG:

1. Install OGG for the source database.
 - i. Extract the OGG installation package for the source database and the following directories appear:

```
drwxr-xr-x install
drwxrwxr-x response
-rwxr-xr-x runInstaller
drwxr-xr-x stage
```

- ii. Install dependencies in response/oggcore.rsp. The OGG response file template is as follows:

```
oracle.install.responseFileVersion=/oracle/install/rspfmt_ogginstall_response_schema
#The installation option, which must reflect the installed Oracle version. Specify ORA11g for installing OGG for Oracle Database 11g.
INSTALL_OPTION=ORA11g
#The location in which OGG is installed.
SOFTWARE_LOCATION=/home/oracle/u01/ggate
#Indicates whether to start the Manager after installation.
START_MANAGER=false
#The port number of the Manager process.
MANAGER_PORT=7839
#The location of the Oracle database.
DATABASE_LOCATION=/home/oracle/u01/app/oracle/product/11.2.0/dbhome_1
#The location that stores the inventory files. This parameter is not required to be configured.
INVENTORY_LOCATION=
#The UNIX group of the inventory directory. In this example, OGG is installed by using the ogg_test Oracle account. You can also create a dedicated account for OGG as necessary.
UNIX_GROUP_NAME=oinstall
```

- iii. Run the following command to install OGG:

```
runInstaller -silent -responseFile {YOUR_OGG_INSTALL_FILE_PATH}/response/oggcore.rsp
```

Note

In this example, OGG is installed in `/home/oracle/u01/ggate` and the installation logs are stored in `/home/oracle/u01/ggate/cfgtoollogs/oui`. The OGG installation is complete when the following message appears in the `silentInstall{time}.log` file:

```
The installation of Oracle GoldenGate Core was successful.
```

- iv. Run the following command and enter `CREATE SUBDIRS` as required to create OGG directories:

```
/home/oracle/u01/ggate/ggsci
```

2. Perform Oracle configurations in the source database.

Navigate to *sqlplus: sqlplus / as sysdba* as the database administrator and complete the following configurations:

```
#Create a tablespace.
create tablespace ATMV datafile '/home/oracle/u01/app/oracle/oradata/uprr/ATMV.dbf' size 100m autoextend
on next 50m maxsize unlimited;
#Create a user named ogg_test. The password is also set to ogg_test.
create user ogg_test identified by ogg_test default tablespace ATMV;
#Grant required privileges to ogg_test.
grant connect,resource,dba to ogg_test;
#Check whether supplemental logging is enabled for the database.
Select SUPPLEMENTAL_LOG_DATA_MIN, SUPPLEMENTAL_LOG_DATA_PK, SUPPLEMENTAL_LOG_DATA_UI, SUP
PLEMENTAL_LOG_DATA_FK, SUPPLEMENTAL_LOG_DATA_ALL from v$database;
#If the result is NO, enable supplemental logging.
alter database add supplemental log data;
alter database add supplemental log data (primary key, unique,foreign key) columns;
#Enable rollback.
alter database drop supplemental log data (primary key, unique,foreign key) columns;
alter database drop supplemental log data;
#Enable all column logging at the database level. Note: Even when all column logging is enabled, only primary
key columns are logged for a delete operation.
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;
#Enable the forced logging mode.
alter database force logging;
#Run the marker_setup.sql script.
@marker_setup.sql
#Run the ddl_setup.sql script.
@ddl_setup.sql
#Run the role_setup.sql script.
@role_setup.sql
#Grant the GGS_GGSUSER_ROLE to ogg_test.
grant GGS_GGSUSER_ROLE to ogg_test;
#Run the ddl_enable.sql script to enable the DDL trigger.
@ddl_enable.sql
#Run the ddl_pin script to improve the performance of the DDL trigger.
@ddl_pin ogg_test
#Run the sequence.sql script.
@sequence.sql
#
alter table sys.seq$ add supplemental log data (primary key) columns;
```

3. Configure the Manager process on the source database.

Start the Oracle GoldenGate Software Command Interface (GGSCI) and perform the following steps:

- i. Run the following command to configure the Manager process:

```
edit params mgr
PORT 7839
DYNAMICPORTLIST 7840-7849
USERID ogg_test, PASSWORD ogg_test
PURGEOLDEXTRACTS ./dirdat/*, USECHECKPOINTS, MINKEEPDAYS 7
LAGREPORTHOURS 1
LAGINFOMINUTES 30
LAGCRITICALMINUTES 45
PURGEDDLHISTORY MINKEEPDAYS 3, MAXKEEPDAYS 7
PURGEMARKERHISTORY MINKEEPDAYS 3, MAXKEEPDAYS 7
```

- ii. Run the following command to start the Manager process. The logs are stored in ggate/dirrpt.

```
start mgr
```

- iii. Run the following command to check whether the Manager process is running:

```
info mgr
```

- iv. Run the following command to view the Manager parameter file:

```
view params mgr
```

4. Configure the Extract process on the source database.

Start the GGSCI and perform the following steps:

- i. Run the following command to configure the Extract process. In the following example, the group name of the process is extract.

```
edit params extractEXTRACT extract
SETENV (NLS_LANG="AMERICAN_AMERICA.AL32UTF8")
DBOPTIONS ALLOWUNUSEDCOLUMN
USERID ogg_test, PASSWORD ogg_test
REPORTCOUNT EVERY 1 MINUTES, RATE
NUMFILES 5000
DISCARDFILE ./dirrpt/ext_test.dsc, APPEND, MEGABYTES 100
DISCARDROLLOVER AT 2:00
WARNLONGTRANS 2h, CHECKINTERVAL 3m
EXTTRAIL ./dirdat/st, MEGABYTES 200
DYNAMICRESOLUTION
TRANLOGOPTIONS CONVERTUCS2CLOBS
TRANLOGOPTIONS RAWDEVICEOFFSET 0
DDL &
INCLUDE MAPPED OBJTYPE 'table' &
INCLUDE MAPPED OBJTYPE 'index' &
INCLUDE MAPPED OBJTYPE 'SEQUENCE' &
EXCLUDE OPTYPE COMMENT
DDLOPTIONS NOCROSSRENAME REPORT
TABLE OGG_TEST.*;
SEQUENCE OGG_TEST.*;
GETUPDATEBEFORES
```

- ii. Run the following command to add an Extract process. Replace extract in the following command with your actual group name.

```
add ext extract,tranlog, begin now
```

- iii. Run the following command to delete an Extract process. In the following example, the process name is DP_TEST.

```
delete ext DP_TEST
```

- iv. Run the following command to create a trail, associate the trail with the Extract group named extract, and set the maximum file size in the trail to 200 megabytes:

```
add extrail ./dirdat/st,ext extract, megabytes 200
```

- v. Run the following command to start the Extract process. The logs are stored in `ggate/dirprt`.

```
start extract extract
```

 **Note** After the Extract process configuration is complete, you can view the changes to the database in the files stored in the `ggate/dirdat` directory.

5. Create a DEFGEN parameter file.

- i. Start the GGSCI in the source database. In GGSCI, run the following command to create a DEFGEN parameter file and copy the file to the `dirdef` directory in the destination database:

```
edit params defgen  
DEFSFILE ./dirdef/ogg_test.def  
USERID ogg_test, PASSWORD ogg_test  
table OGG_TEST. *;
```

- ii. Run the following command from the shell to create a DEFGEN parameter file named `ogg_test.def`:

```
./defgen paramfile ./dirprm/defgen.prm
```

6. Install and configure OGG in the destination database.

- i. Extract the OGG installation package to the destination database.
- ii. Copy the `dirdef/ogg_test.def` file in the source database to `dirdef` of the destination database.
- iii. Start the GGSCI and run the following command to create the default directories of OGG:

```
create subdirs
```

- iv. Run the following command to configure the Manager process:

```
edit params mgr  
PORT 7839  
DYNAMICPORTLIST 7840-7849  
PURGEOLDEXTRACTS ./dirdat/*, USECHECKPOINTS, MINKEEPDAYS 7  
LAGREPORHOURS 1  
LAGINFOMINUTES 30  
LAGCRITICALMINUTES 45  
PURGEDDLHISTORY MINKEEPDAYS 3, MAXKEEPDAYS 7  
PURGEMARKERHISTORY MINKEEPDAYS 3, MAXKEEPDAYS 7
```

- v. Run the following command to start the Manager process:

```
start mgr
```

7. Configure a data pump in the source database.

Start the GGSCI and perform the following steps:

- i. Run the following command to configure a data pump:

```
edit params pump
EXTRACT pump
RMTHOST xx.xx.xx.xx, MGRPORT 7839, COMPRESS
PASSTHRU
NUMFILES 5000
RMTTRAIL ./dirdat/st
DYNAMICRESOLUTION
TABLE OGG_TEST.*;
SEQUENCE OGG_TEST.*;
```

- ii. Run the following command to create a data-pump Extract process. The process reads from the specified trail.

```
add ext pump,exttrailsources ./dirdat/st
```

- iii. Run the following command to create a trail and set the maximum file size in the trail to 200 megabytes:

```
add rmttrail ./dirdat/st,ext pump,megabytes 200
```

- iv. Run the following command to start the data pump:

```
start pump
```

 **Note** After the data pump is started, you can view the trail files in the dirdat directory of the destination database.

8. Install and configure the DataHub agent for OGG.

- i. Run the following command to configure the *JAVA_HOME* and *LD_LIBRARY_PATH* environment variables and specify the configurations in the *~/ .bash_profile*:

```
export JAVA_HOME=/xxx/xxx/jrexx
export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:$JAVA_HOME/lib/amd64:$JAVA_HOME/lib/amd64/server
```

- ii. After the environment variables are configured, restart the Manager process in the destination database.
- iii. Download the [DataHub agent for OGG](#) and extract the installation package.
- iv. Modify the *javae.properties* and *log4j.properties* files in the *conf* sub-directory of the installation directory and replace *{YOUR_HOME}* with the target path of the extracted files:

```
gg.handlerlist=ggdatahub
gg.handler.ggdatahub.type=com.aliyun.odps.ogg.handler.datahub.DatahubHandler
gg.handler.ggdatahub.configureFileName={YOUR_HOME}/datahub-ogg-plugin/conf/configure.xml
goldengate.userexit.nochkpt=false
goldengate.userexit.timestamp=utc
gg.classpath={YOUR_HOME}/datahub-ogg-plugin/lib/*
gg.log.level=debug
jvm.bootoptions=-Xmx512m -Dlog4j.configuration=file:{YOUR_HOME}/datahub-ogg-plugin/conf/log4j.properties -Djava.class.path=ggjava/ggjava.jar
```

- v. Modify the *configure.xml* file in the *conf* sub-directory of the installation directory as follows:

```
<? xml version="1.0" encoding="UTF-8"? >
<configure>
  <defaultOracleConfigure>
    <!--(Required) The Oracle database system identifier (SID) -->
```

```
<!--(Required) The Oracle database system identifier (SID).-->
<sid>100</sid>
<!--The schema of the Oracle table, which can be overwritten by oracleSchema in the column mapping
s. At least one of them must be specified.-->
<schema>ogg_test</schema>
</defaultOracleConfigure>
<defaultDatahubConfigure>
<!--(Required) The endpoint of DataHub.-->
<endPoint>YOUR_DATAHUB_ENDPOINT</endPoint>
<!--The DataHub project, which can be overwritten by datahubProject in the column mappings. At lea
st one of them must be specified.-->
<project>YOUR_DATAHUB_PROJECT</project>
<!--The AccessKey ID for accessing DataHub, which can be overwritten by datahubAccessId in the colu
mn mappings. At least one of them must be specified.-->
<accessId>YOUR_DATAHUB_ACCESS_ID</accessId>
<!--The AccessKey Secret for accessing DataHub, which can be overwritten by datahubAccessKey in th
e column mappings. At least one of them must be specified.-->
<accessKey>YOUR_DATAHUB_ACCESS_KEY</accessKey>
<!--The column in DataHub that indicates the data update type, which can be overwritten by ctypeCol
umn in the column mappings.-->
<ctypeColumn>optype</ctypeColumn>
<!-- The column in DataHub that indicates the data update time, which can be overwritten by ctimeCo
lumn in the column mappings.-->
<ctimeColumn>readtime</ctimeColumn>
<!-- The column in DataHub that indicates the sequence number of the updated data, which can be ov
erwritten by cidColumn in the column mappings. The sequence number increases as more data are updat
ed, but may not be consecutive.-->
<cidColumn>record_id</cidColumn>
</defaultDatahubConfigure>
<!--The approach to handling errors. If an error occurs, the system either ignores the error and continue
s running or retries the operation repeatedly.-->
<!--(Optional) The maximum number of records operated at one time. Default value: 1000.-->
<batchSize>1000</batchSize>
<!--(Optional) The format that the timestamp is converted into. Default: yyyy-MM-dd HH:mm:ss.-->
<defaultDateFormat>yyyy-MM-dd HH:mm:ss</defaultDateFormat>
<!--(Optional) Indicates whether the system needs to ignore dirty records. Default value: false.-->
<dirtyDataContinue>>true</dirtyDataContinue>
<!--(Optional) The dirty record file name. Default value: datahub_ogg_plugin.dirty-->
<dirtyDataFile>datahub_ogg_plugin.dirty</dirtyDataFile>
<!--(Optional) The maximum size of the dirty record file. Unit: MB. Default value: 500.-->
<dirtyDataFileMaxSize>200</dirtyDataFileMaxSize>
<!--(Optional) The maximum number of retries if an error occurs. -1: Unlimited. 0: No retries. n: The num
ber of retries. Default value: -1.-->
<retryTimes>0</retryTimes>
<!--(Optional) The interval between retries. Unit: milliseconds. Default value: 3000.-->
<retryInterval>4000</retryInterval>
<!--(Optional) The checkpoint file name. Default value: datahub_ogg_plugin.chk.-->
<checkPointFileName>datahub_ogg_plugin.chk</checkPointFileName>
< mappings>
< mapping>
<!--The schema of the Oracle table.-->
<oracleSchema></oracleSchema>
<!--(Required) The Oracle table name.-->
<oracleTable>t_person</oracleTable>
<!--The DataHub project name.-->
<datahubProject></datahubProject>
<!--The AccessKey ID for accessing DataHub.-->
<datahubAccessId></datahubAccessId>
```

```

<!--The AccessKey Secret for accessing DataHub.-->
<datahubAccessKey></datahubAccessKey>
<!--(Required) The DataHub topic name.-->
<datahubTopic>t_person</datahubTopic>
<ctypeColumn></ctypeColumn>
<ctimeColumn></ctimeColumn>
<cidColumn></cidColumn>
<columnMapping>
  <!--
    src: (Required) The column names in the Oracle table.
    dest: (Required) The column names in the DataHub topic.
    destOld: (Optional) The DataHub topic column that records the data before it is updated.
    isShardColumn: (Optional) Indicates whether the shard ID is generated based on the hash key value, which can be overwritten by shardId. Default value: false.
    isDateFormat: Indicates whether the timestamp is converted into a string based on dateFormat. Default value: true. If you set the value to false, the data type in the source database must be long.
    dateFormat: The format that the timestamp is converted into. If this parameter is left blank, the default format is used.
  -->
  <column src="id" dest="id" isShardColumn="true" isDateFormat="false" dateFormat="yyyy-MM-dd HH:mm:ss"/>
  <column src="name" dest="name" isShardColumn="true"/>
  <column src="age" dest="age"/>
  <column src="address" dest="address"/>
  <column src="comments" dest="comments"/>
  <column src="sex" dest="sex"/>
  <column src="temp" dest="temp" destOld="temp1"/>
</columnMapping>
<!--(Optional) The ID of the shard prioritized to be written into.-->
<shardId>1</shardId>
</mapping>
</mappings>
</configure>

```

- vi. Run the following command in GGSCI to start the DataHub writer:

```

edit params dhwriter
extract dhwriter
getEnv (JAVA_HOME)
getEnv (LD_LIBRARY_PATH)
getEnv (PATH)
CUSEREXIT ./libggjava_ue.so CUSEREXIT PASSTHRU INCLUDEUPDATEBEFORES, PARAMS "{YOUR_HOME}/datahub-ogg-plugin/conf/javaue.properties"
sourcedefs ./dirdef/ogg_test.def
table OGG_TEST.*;

```

- vii. Run the following command to add a DataHub writer:

```
add extract dhwriter, extrailsources ./dirdef/st
```

- viii. Run the following command to start the writer:

```
start dhwriter
```

Use case

For example, you have an Oracle table that stores order information. The table has three columns. The column names are oid, pid, and num, which indicate order ID, product ID, and product quantity. You can synchronize incremental data to DataHub by using the DataHub agent for OGG. The steps are as follows:

Note Before performing incremental data synchronization, you must synchronize existing data from the source table to MaxCompute by using DataX.

1. Create a topic in DataHub. The schema of the topic is as follows:

```
string record_id, string optype, string readtime, bigint oid_before, bigint oid_after, bigint pid_before, bigint pid_after, bigint num_before, bigint num_after
```

2. Make sure that you have completed the deployment of the DataHub agent for OGG. Then configure the column mappings as follows:

```
<ctypeColumn>optype</ctypeColumn>
<ctimeColumn>readtime</ctimeColumn>
<cidColumn>record_id</cidColumn>
<columnMapping>
  <column src="oid" dest="oid_after" destOld="oid_before" isShardColumn="true"/>
  <column src="pid" dest="pid_after" destOld="pid_before"/>
  <column src="num" dest="num_after" destOld="num_before"/>
</columnMapping>
```

Note The optype parameter indicates the type of the data update. Valid values of the optype parameter are I, D, and U, which represent an insert, delete, and update operation, respectively. The readtime parameter indicates the time of the data update.

3. When the agent can run properly, data updates are synchronized from the source table to DataHub.

6.6. Data synchronization

6.6.1. Overview

You can synchronize real-time data from DataHub to other data warehouses by using DataConnectors so that you can analyze and process historical data.

The following topics describe how to synchronize data from DataHub to MaxCompute.

6.6.2. Synchronize data to MaxCompute

6.6.2.1. Create a DataConnector

This topic describes how to create a DataConnector to synchronize data from DataHub to MaxCompute.

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click View in the Operate column. On the project details page, find the target topic and click View in the Operate column.
3. On the topic details page, click **Connector**. In the Create connector dialog box, select the data warehouse to which data is synchronized.
4. In the Create connector dialog box, set relevant parameters and click **Create**.

 **Note**

The following table describes the parameters of the DataConnector for synchronizing data from DataHub to MaxCompute.

Parameter	Description
Project Name	The name of the MaxCompute project to which data in the topic is synchronized.
Table Name	The name of the MaxCompute table to which data in the topic is synchronized.
AccessID and AccessKey	The AccessKey pair used to access MaxCompute. The AccessKey pair must belong to a RAM user that has CreateInstance, Desc, and Alter permissions on the MaxCompute table.
Partition Mode	The method used to create partitions. Valid values: SYSTEM_TIME, EVENT_TIME, USER_DEFINE, and META_TIME. If you select SYSTEM_TIME, partitions are created based on the recording time. If you select EVENT_TIME, partitions are created based on the value of the event_time field. When you create the topic, you must define a field named event_time for the topic and set its data type to TIMESTAMP. The value of the event_time field must be accurate to microseconds. If you select USER_DEFINE, partitions are created based on the user-defined partition key.
Partition Config	The format of the time based on which partitions are created. This parameter takes effect only when you set the Partition Mode parameter to SYSTEM_TIME, EVENT_TIME, or META_TIME.
Time Range	The interval of creating partitions. This parameter takes effect only when you set the Partition Mode parameter to SYSTEM_TIME, EVENT_TIME, or META_TIME. The minimum value is 15 minutes.
Timezone	The time zone of the time based on which partitions are created. This parameter takes effect only when you set the Partition Mode parameter to SYSTEM_TIME, EVENT_TIME, or META_TIME.
Start Time	The time when data synchronization starts.

6.6.2.2. View data synchronization details

This section describes how to view data synchronization details after a DataConnector is created.

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click **View** in the Operate column.
3. On the project details page, find the target topic and click **View** in the Operate column.
4. On the topic details page, click the **Connector** tab. On the Connector tab, find the target DataConnector and click **View** in the Operate column.

 **Notice** You can restart or stop a DataConnector. Exercise caution when you perform the operations.

6.7. Metric statistics

This topic describes how to view the metric statistics of a topic in DataHub.

In the DataHub console, you can view the metric statistics of topics in quasi-real-time, such as QPS and throughput. The following metrics are available:

- Read and write QPS
 - Read and write records per second (RPS)
 - Read and write throughput, measured in KB per second
 - Read and write latency, measured in microseconds per request
1. Log on to the DataHub console.
 2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click **View** in the Operate column.
 3. On the project details page, find the target topic and click **View** in the Operate column.
 4. On the topic details page, click the **Metric Statistics** tab.

You can view the metric statistics for a specified time range.

6.8. Data subscription

6.8.1. Overview

Resumable consumption is required in scenarios where you consume data in DataHub topics and want to resume the consumption from the time when your application fails. If you need to resume consumption, you must save the current consumption offset and make sure that the service for saving consumption offsets supports high availability. This increases the complexity of developing applications. The subscription feature of DataHub supports saving consumption offsets to the server to solve the preceding problem. You only need to enable this feature and add a few lines of code to your application to obtain a consumption offset maintenance service with high availability.

In addition, the subscription feature allows you to reset consumption offsets. This ensures that the data can be consumed at least once. For example, if an error occurs when your application processes the data consumed in a specific time period and you need to consume the data again, you can reset the consumption offset without restarting the application. Your application automatically consumes data from the specified consumption offset.

6.8.2. Create a subscription

You can create subscriptions only in the DataHub console. Make sure that your account is authorized to subscribe to topics of the specified project.

Perform the following steps to create a subscription:

1. Log on to the DataHub console.
2. In the left-side navigation pane, click **Project Manager**. On the Project List page, find the target project and click **View** in the Operate column. On the project details page, find the target topic and click **View** in the Operate column.
3. On the topic details page, click **Subscription**. In the Create subscription dialog box, set relevant parameters and click **Create**.
4. After the subscription is created, click the **Subscription List** tab on the topic details page to view the subscriptions of the topic.

-  **Note** You can click **Reset** or **Delete** in the Operate column of a subscription.
- **Reset**: resets the consumption offset of the subscription to the required time. Specify the time in the *mm-dd-yyyy HH:MM:SS* format.
 - **Delete**: permanently deletes the subscription, including all consumption offsets that are saved for the subscription.

6.8.3. Use case

The subscription feature allows you to save consumption offsets. You can use the read and write capabilities of DataHub with the capability of saving consumption offsets in scenarios where you must save consumption offsets after data is read.

The following sample code is used for reference only.

```
// The following sample code consumes data from a saved consumption offset and submit consumption offsets during consumption.
public void offset_consumption(int maxRetry) {
    String endpoint = "<YourEndPoint>";
    String accessId = "<YourAccessId>";
    String accessKey = "<YourAccessKey>";
    String projectName = "<YourProjectName>";
    String topicName = "<YourTopicName>";
    String subId = "<YourSubId>";
    String shardId = "0";
    List<String> shardIds = Arrays.asList(shardId);
    // Create a DataHub client.
    DatahubClient datahubClient = DatahubClientBuilder.newBuilder()
        .setDatahubConfig(
            new DatahubConfig(endpoint,
                // Specify whether to enable binary data transmission. The server of V2.12 or later supports binary data transmission.
                new AliyunAccount(accessId, accessKey), true))
        .build();
    RecordSchema schema = datahubClient.getTopic(projectName, topicName).getRecordSchema();
    OpenSubscriptionSessionResult openSubscriptionSessionResult = datahubClient.openSubscriptionSession(projectName, topicName, subId, shardIds);
    SubscriptionOffset subscriptionOffset = openSubscriptionSessionResult.getOffsets().get(shardId);
    // 1. Obtain the cursor of the record at the current consumption offset. If the record expired or the record is not consumed, obtain the cursor of the first record within the TTL of the topic.
    String cursor = "";
    // If the sequence number is smaller than 0, the record is not consumed.
    if (subscriptionOffset.getSequence() < 0) {
        // Obtain the cursor of the first record within the TTL of the topic.
        cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.OLDEST).getCursor();
    } else {
        // Obtain the cursor of the next record.
        long nextSequence = subscriptionOffset.getSequence() + 1;
        try {
            // If the SeekOutOfRangeException error is returned after you obtain the cursor based on the sequence number, the record expired.
            cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.SEQUENCE, nextSequence).getCursor();
        } catch (SeekOutOfRangeException e) {
            // Obtain the cursor of the first record within the TTL of the topic.
            cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.OLDEST).getCursor();
        }
    }
}
```

```
}
}
// 2. Read records and save consumption offsets. In this example, you read tuple records and save consumption of
// offsets each time 1,000 records are read.
long recordCount = 0L;
// Read 1,000 records each time.
int fetchNum = 1000;
int retryNum = 0;
int commitNum = 1000;
while (retryNum < maxRetry) {
    try {
        GetRecordsResult getRecordsResult = datahubClient.getRecords(projectName, topicName, shardId, schema, c
        ursor, fetchNum);
        if (getRecordsResult.getRecordCount() <= 0) {
            // If no records can be read, pause the thread for 1s and continue to read records.
            System.out.println("no data, sleep 1 second");
            Thread.sleep(1000);
            continue;
        }
        for (RecordEntry recordEntry : getRecordsResult.getRecords()) {
            // Consume data.
            TupleRecordData data = (TupleRecordData) recordEntry.getRecordData();
            System.out.println("field1:" + data.getField("field1") + "\t"
                + "field2:" + data.getField("field2"));
            // Save the consumption offset after the data is consumed.
            recordCount++;
            subscriptionOffset.setSequence(recordEntry.getSequence());
            subscriptionOffset.setTimestamp(recordEntry.getSystemTime());
            // commit offset every 1000 records
            if (recordCount % commitNum == 0) {
                // Submit the consumption offset.
                Map<String, SubscriptionOffset> offsetMap = new HashMap<>();
                offsetMap.put(shardId, subscriptionOffset);
                datahubClient.commitSubscriptionOffset(projectName, topicName, subId, offsetMap);
                System.out.println("commit offset successful");
            }
        }
        cursor = getRecordsResult.getNextCursor();
    } catch (SubscriptionOfflineException | SubscriptionSessionInvalidException e) {
        // The subscription session is exited. The Offline exception indicates that the subscription is offline. The Sessi
        onChange exception indicates that the subscription is consumed by other clients.
        e.printStackTrace();
        throw e;
    } catch (SubscriptionOffsetResetException e) {
        // The consumption offset is reset. You must obtain the version information of the consumption offset again.
        SubscriptionOffset offset = datahubClient.getSubscriptionOffset(projectName, topicName, subId, shardIds).g
        etOffsets().get(shardId);
        subscriptionOffset.setVersionId(offset.getVersionId());
        // After the consumption offset is reset, you must obtain the cursor of the record at the consumption offset ag
        ain. The method for obtaining the cursor depends on the method of resetting the consumption offset.
        // If both the sequence number and timestamp are specified to reset the consumption offset, you can obtain t
        he cursor based on the sequence number or the timestamp.
        // If only the sequence number is specified to reset the consumption offset, you can obtain the cursor only bas
        ed on the sequence number.
        // If only the timestamp is specified to reset the consumption offset, you can obtain the cursor only based on t
        he timestamp.
        // Generally, preferentially obtain the cursor based on the sequence number. If the cursor failed to be obtaine
        d based on the sequence number or the timestamp, obtain the cursor of the earliest record.
    }
}
```

```
    cursor = null;
    if (cursor == null) {
        try {
            long nextSequence = offset.getSequence() + 1;
            cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.SEQUENCE, nextSequence).getCursor();
            System.out.println("get cursor successful");
        } catch (DatahubClientException exception) {
            System.out.println("get cursor by SEQUENCE failed, try to get cursor by SYSTEM_TIME");
        }
    }
    if (cursor == null) {
        try {
            cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.SYSTEM_TIME, offset.getTimestamp()).getCursor();
            System.out.println("get cursor successful");
        } catch (DatahubClientException exception) {
            System.out.println("get cursor by SYSTEM_TIME failed, try to get cursor by OLDEST");
        }
    }
    if (cursor == null) {
        try {
            cursor = datahubClient.getCursor(projectName, topicName, shardId, CursorType.OLDEST).getCursor();
            System.out.println("get cursor successful");
        } catch (DatahubClientException exception) {
            System.out.println("get cursor by OLDEST failed");
            System.out.println("get cursor failed!!");
            throw e;
        }
    }
} catch (LimitExceededException e) {
    // limit exceed, retry
    e.printStackTrace();
    retryNum++;
} catch (DatahubClientException e) {
    // other error, retry
    e.printStackTrace();
    retryNum++;
} catch (Exception e) {
    e.printStackTrace();
    System.exit(-1);
}
}
```

 **Note**

- When you start the application for the first time, your application consumes data from the earliest record. During the running of the application, you can refresh the Subscription List tab in the console.
- If you reset the consumption offset by clicking Reset in the console during the consumption, your application automatically detects the change of the consumption offset and consumes data from the specified consumption offset. When the application catches `OffsetResetedException`, the application calls the `getSubscriptionOffset` method to query the latest consumption offset from the server. Then, the application can consume data from the latest consumption offset.
- Note that a shard in a subscription cannot be consumed by multiple threads or processes at the same time. Otherwise, the consumption offset submitted by a thread is overwritten by that submitted by another thread and the server cannot determine to which thread the saved consumption offset belongs. In this case, the server throws `OffsetSessionChangedException`. We recommend that you exit the subscription session to check whether data is repeatedly consumed if this exception is caught.

6.9. Collaborative consumption

6.9.1. Note

DataHub-client-library encapsulates the Java SDK and integrates the consumer for collaborative consumption and the producer for distributing data evenly among shards.

6.9.2. Overview

Offset-based data consumption

The offset-based data consumption feature allows you to save consumption offsets to the server. A consumption offset consists of the sequence number of a record and the timestamp when the record is written to DataHub.

You can create a subscription for a topic and submit the consumption offset to the server after specific data is consumed. When your application starts the next time, the application can obtain the consumption offset from the server and consume data from the next record. The consumption offsets must be saved on the server so that your application can consume data from a submitted consumption offset after shards are reallocated. This is a prerequisite for collaborative consumption.

You do not need to manually submit consumption offsets in the consumer. You only need to specify the interval of submitting consumption offsets in the configurations of the consumer. The system considers that the previous records are consumed when it reads records. If the interval of submitting consumption offsets is exceeded, the system submits a consumption offset again. If the consumption offset fails to be submitted and your application is interrupted, the consumption offset may fail to be submitted in time. In this case, your application may repeatedly consume specific data.

Collaborative consumption

The collaborative consumption feature automatically allocates shards when multiple consumers consume a topic at the same time. This feature simplifies the data processing of clients.

 **Note** Manual shard allocation is difficult because multiple consumers may reside on different machines. If multiple consumers that subscribe to the same topic are in the same consumer group, a shard can be allocated to only one consumer in the consumer group.

Example:

Assume that A, B, and C are three consumer instances and the topic has 10 shards.

1. When the consumer instance A is started at first, 10 shards are allocated to it.
2. When the other two consumer instances are started, the shards are reallocated in the following way: four to A, three to B, and three to C.
3. When one of the shards consumed by the consumer instance A is split into two and the two shards are released after consumption, the shards are reallocated in the following way: four to A, four to B, and three to C.
4. When the consumer instance C is stopped, the shards are reallocated in the following way: six to A and five to B.

Heartbeat

You must use the heartbeat feature to notify the server of the status of consumer instances. If the server has not received heartbeats from a consumer instance after the specified interval, the server considers that the consumer instance is stopped. When the status of a consumer instance changes, the server reallocates shards. The server returns the new allocation plan in heartbeat requests. Therefore, the client takes time to detect reallocation of shards.

6.9.3. Maven dependencies and JDK

Maven dependencies

```
<dependency>
  <groupId>com.aliyun.datahub</groupId>
  <artifactId>datahub-client-library</artifactId>
  <version>1.0.6-public</version>
</dependency>
```

JDK

```
jdk: >= 1.7
```

6.9.4. Use case

The following sample code is for reference only.

Initialize the producer

```
String endpoint = "http://dh-cn-hangzhou.aliyuncs.com";
String accessId = "<YourAccessKeyId>";
String accessKey = "<YourAccessKeySecret>";
String projectName = "<YourProjectName>";
String topicName = "<YourTopicName>";
ProducerConfig config = new ProducerConfig(endpoint, accessId, accessKey);
Producer producer = new Producer(projectName, topicName, config);
```

Write data to DataHub

```
RecordSchema schema = new RecordSchema();
schema.addField(new Field("field1", FieldType.STRING));
schema.addField(new Field("field2", FieldType.BIGINT));
List<RecordEntry> recordEntries = new ArrayList<>();
for (int cnt = 0; cnt < 10; ++cnt) {
    RecordEntry entry = new RecordEntry();
    entry.addAttribute("key1", "value1");
    entry.addAttribute("key2", "value2");
    TupleRecordData data = new TupleRecordData(schema);
    data.setField("field1", "testValue");
    data.setField("field2", 1);
    entry.setRecordData(data);
    recordEntries.add(entry);
}
int maxRetry = 3;
while (true) {
    try {
        producer.send(records, maxRetry);
        break;
    } catch (MalformedRecordException e) {
        // malformed RecordEntry
    } catch (InvalidParameterException e) {
        // invalid param
    } catch (ResourceNotFoundException e) {
        // project, topic or shard not found, sometimes caused by split/merge shard
    } catch (DatahubClientException e) {
        // network or other exceptions exceeded retry limit
    }
}
// close before exit
producer.close();
```

Initialize the consumer

```
String endpoint = "http://dh-cn-hangzhou.aliyuncs.com";
String accessId = "<YourAccessKeyId>";
String accessKey = "<YourAccessKeySecret>";
String projectName = "<YourProjectName>";
String topicName = "<YourTopicName>";
String SubId = "<YourSubscriptionId>";
// 1. If you need to use the collaborative consumption feature, specify the subscription ID.
ConsumerConfig config = new ConsumerConfig(endpoint, accessId, accessKey);
Consumer consumer = new Consumer(projectName, topicName, SubId, config);
// 2. If you need to use the offset-based data consumption feature instead of the collaborative consumption feature
, specify the subscription ID and the shards to be read by the consumer.
List<String> assignment = Arrays.asList("0", "1", "2");
ConsumerConfig config = new ConsumerConfig(endpoint, accessId, accessKey);
Consumer consumer = new Consumer(projectName, topicName, SubId, assignment, config);
// 3. If you do not need to use the collaborative consumption feature nor the offset-based data consumption feature,
specify the subscription ID, the shards to be read by the consumer, and the consumption offset.
Map<String, Offset> offsetMap = new HashMap<>();
// If both the sequence number and timestamp are specified but the sequence number is invalid, obtain the cursor
based on the timestamp.
offsetMap.put("0", new Offset(100, 1548573440756L));
// If only the sequence number is specified, obtain the cursor based on the sequence number.
offsetMap.put("1", new Offset().setSequence(1));
// If only the timestamp is specified, obtain the cursor based on the timestamp.
offsetMap.put("2", new Offset().setTimestamp(1548573440756L));
ConsumerConfig config = new ConsumerConfig(endpoint, accessId, accessKey);
Consumer consumer = new Consumer(projectName, topicName, SubId, offsetMap, config);
```

Read data from DataHub

```
int maxRetry = 3;
boolean stop = false;
while (! stop) {
    try {
        while (true) {
            RecordEntry record = consumer.read(maxRetry);
            if (record != null) {
                TupleRecordData data = (TupleRecordData) record.getRecordData();
                System.out.println("field1:" + data.getField(0) + ", field2:" + data.getField("field2"));
            }
        }
    } catch (SubscriptionSessionInvalidException | SubscriptionOffsetResetException e) {
        // subscription exception, will not recover
        // print some log or just use a new consumer
        consumer.close();
        consumer = new Consumer(TEST_PROJECT, TEST_TOPIC, TEST_SUB_ID, config);
    } catch (ResourceNotFoundException | InvalidParameterException e) {
        // - project, topic, shard, subscription not found
        // - seek out of range
        // - sometimes shard operation cause ResourceNotFoundException
        // should make sure if resource exists, print some log or just exit
    } catch (DatahubClientException e) {
        // - network or other exception exceed retry limit
        // can just sleep and retry
    }
}
// close before exit
consumer.close();
```

6.9.5. Usage notes

A consumer or producer cannot access DataHub by using multiple threads. If you need to use multiple threads, specify a different consumer or producer for each thread.

7. Quick BI

7.1. What is Quick BI?

This topic describes the concept and features of Quick BI.

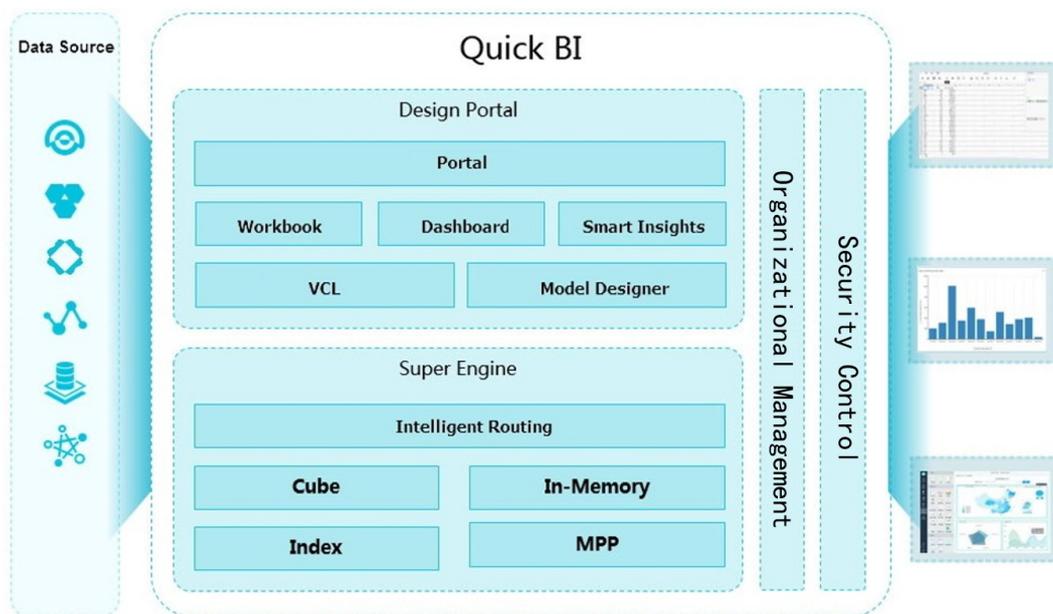
Quick BI is a flexible and lightweight self-service Business Intelligence (BI) tool based on cloud computing. It supports a wide range of data sources:

- MaxCompute (formerly known as ODPS) and ApsaraDB for RDS
- User-created MySQL and SQL Server databases that are hosted on ECS
- VPC data sources

Quick BI analyzes large amounts of data in real time and returns results within seconds. You do not need to preprocess the data. Quick BI analyzes terabytes of incremental data on a daily basis.

With an intelligent data modeling tool and a variety of visual chart tools, Quick BI significantly reduces data acquisition costs and makes it easier for you to use Quick BI features. This allows you to easily complete data analysis, self-service data acquisition, business data query, and report making.

Architecture



7.2. Log on to the Quick BI console

This topic describes how to log on to the Quick BI console.

Prerequisites

- The IP address or domain name of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console. The access address of the Apsara Uni-manager Management Console is `http://IP address or domain name of the Apsara Uni-manager Management Console/manage`.
- We recommend that you use the Google Chrome browser.

Context

If you are using a RAM account, you can use the domain name to log on to the Quick BI console. If you are using an Apsara Stack tenant account, log on to the Quick BI console by performing the following steps:

Procedure

1. In the address bar of the browser, enter the URL of the Apsara Uni-manager Management Console and press the Enter key.
2. Enter your username and password.

Obtain the username and password that are used to log on to the console from the operations administrator.

 **Note** If this is the first time you log on to the Apsara Uni-manager Management Console, you must change your password as prompted. For security reasons, your password must meet the minimum complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters.
- Digits.
- Special characters such as exclamation points (!), at signs (@), number signs (#), dollar signs (\$), and percent signs (%).

3. Click **Login** to go to the homepage of Apsara Uni-manager Management Console.
4. In the top navigation bar, choose **Product > Big Data** and click **Quick BI**.
5. Select an organization and a region from the drop-down lists and click **Quick BI** or **Quick BI Console** to go to the Quick BI system.
 - You can click **Quick BI** to go to the product page.
 - You can click **Quick BI Console** to go to the settings page. You can manage organization members and workspaces on this page.

 **Note** If you use this method, the Apsara Stack tenant account `quickbi_admin@aliyun.com` is used to log on to the Quick BI console no matter which department you select.

7.3. Access mode

Quick BI can work in either development mode or access mode. You can view reports available to you only when Quick BI works in access mode, which provides higher data access efficiency and better user experience.

- You can view reports on the **Common**, **Favorites**, **Shared with Me**, and **Footprint** tabs, and in the **Business Navigation** section. You can click the **More** icon in the upper-right corner of a tab or section to show all reports. You can click **My Applications** to go to the destination application module and view its data.
 - You can view the reports that were accessed in the last 30 days on the **Footprint** tab.
 - You can view the top four report categories with the most visits in the **Business Navigation** section.

 **Note** In access mode, the configuration panel is unavailable.

- In development mode, only developers and analysts can edit reports in workspaces. Visitors cannot access the development mode.

7.4. Data modeling

7.4.1. Data modeling

Data modeling visualizes data and allows you to quickly identify and extract information. It also helps you make correct decisions based on data trends.

Steps of data modeling:

1. Create a data source.
2. Create a dataset.

7.4.2. Data sources

7.4.2.1. Overview

Datasets, workbooks, dashboards, and BI portals are all created based on data sources. Quick BI supports both cloud data sources and user-created data sources.

Cloud data sources include:

- MaxCompute
- MySQL
- SQL Server
- AnalyticDB for MySQL 2.0
- HybridDB for MySQL
- AnalyticDB for PostgreSQL
- PostgreSQL
- PPAS
- Data Lake Analytics
- Hive
- DRDS
- Presto
- AnalyticDB for MySQL 3.0
- PolarDB for MySQL

 **Note** VPCs only support MySQL and SQL Server data sources.

User-created data sources include:

- MySQL
- SQL Server
- PostgreSQL
- Oracle
- Hive
- Vertica
- IBM DB2 LUW
- SAP IQ (SybaseIQ)
- SAP HANA
- Presto

 **Note** You cannot check SQL Server data sources by using views.

7.4.2.2. Cloud data sources

7.4.2.2.1. Add the IP addresses of a Quick BI cluster to a database whitelist

This topic describes how to add the IP addresses of a Quick BI cluster to a database whitelist.

Prerequisites

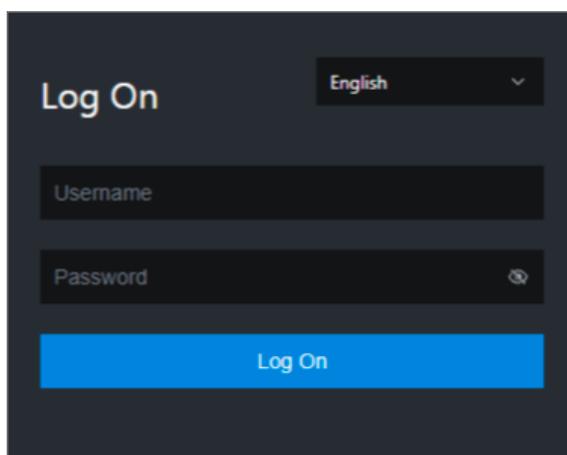
The Quick BI service is purchased.

Context

Before you add some data sources in the Quick BI console, you must first query available IP addresses of machines and add them to a whitelist of ApsaraDB RDS.

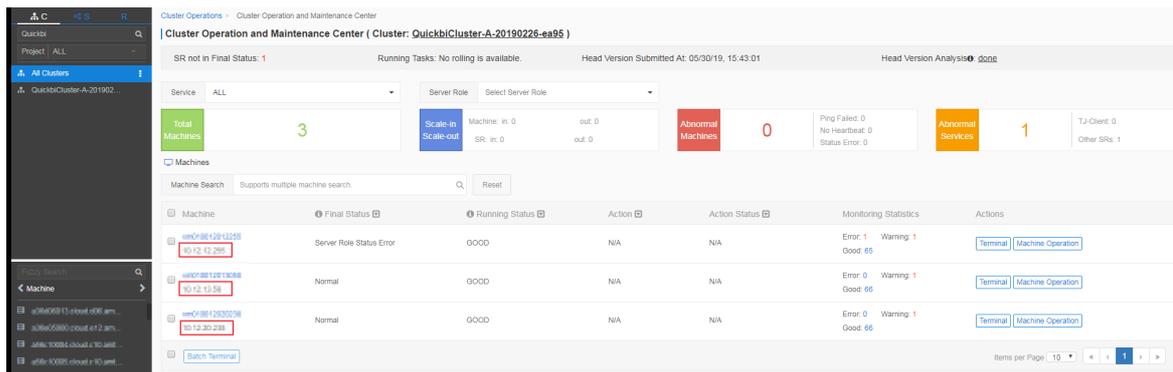
Procedure

1. Log on to the Apsara Infrastructure Management Framework console. Make sure that you have obtained the URL of the Apsara Uni-manager Operations Console, and the username and password used for logging on to the console from a deployment engineer or administrator.
 - i. Open a browser, enter the URL in the address bar, and press Enter. The URL is in the format of *ops.asconsole.intranet-domain-id.com*.



 **Note** We recommend that you use the Google Chrome browser. You can select a language from the drop-down list in the upper-right corner of the page.

- ii. Enter your username and password. If you log on to the Apsara Uni-manager Operations Console for the first time, you must change the password of your username.
 For security concerns, your password must contain the following characters:
 - Letters
 - Digits
 - Special characters, such as exclamation points (!), at signs (@), number signs (#), dollar signs (\$), and percent signs (%)
 - The password must be 10 to 20 characters in length.
 - iii. Click **Log On**.
 - iv. In the top navigation bar, click **O&M**.
 - v. In the left-side navigation pane, choose **Product Management > Products**.
 - vi. In the Apsara Stack O&M section, choose **Basic O&M > Apsara Infrastructure Management Framework**.
 - vii. In the left-side navigation pane of Infra. Operation Platform, choose **Operations > Cluster Operations**.
2. In the search box, select your project or enter the name of the Quick BI cluster. Find the cluster whose configurations you want to query. Click the cluster name or click **Operations** in the **Actions** column that corresponds to the cluster. The **Cluster Details** page appears.
 3. View the IP addresses of the Quick BI cluster on the **Machines** tab.



What's next

Add the IP addresses to an ApsaraDB RDS whitelist.

Before you add the IP addresses to an ApsaraDB RDS whitelist, change the last octet of all the addresses to 0/24. For example, if an IP address is 10.10.10.10, change it to 10.10.10.0/24. For more information about how to add an IP address to an ApsaraDB RDS whitelist, see "Configure a whitelist" in ApsaraDB RDS User Guide.

7.4.2.2.2. Add a MaxCompute data source

This topic describes how to add a MaxCompute data source.

Prerequisites

- A MaxCompute (formerly known as ODPS) project is created in the MaxCompute console.
- Parameter settings for connecting to the MaxCompute data source are obtained.

Procedure

1. [Create a data source](#).
2. Click **MaxCompute**. In the **Add MaxCompute Database** dialog box, configure the following

parameters.

3. Configure the parameters for connecting to the data source.

Parameter	Description
Name	The name of the data source.
Database Address	The default database address.
Project Name	The name of the project.
AccessKey ID	The AccessKey ID used to identify a visitor.
AccessKey Secret	The AccessKey secret is used to encrypt the signature string on the client and decrypt the signature string on the server for authentication. Keep the AccessKey secret confidential.

4. Click **Test Connection** to perform a data source connectivity test.



5. After the database passes the connectivity test, click **OK**. After the data source is added, the Data Sources page appears. Tables under the data source are listed on the right side of the page.

7.4.2.2.3. Add an ApsaraDB RDS for MySQL data source

This topic describes how to add an ApsaraDB RDS for MySQL data source.

Prerequisites

1. An ApsaraDB RDS for MySQL instance is created. For information about how to create an ApsaraDB for RDS instance, see "Create an instance" in ApsaraDB for RDS User Guide.
2. Available IP addresses of machines are obtained and added to a whitelist of the ApsaraDB for RDS instance. For information about how to query the IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add the IP addresses to a whitelist of the ApsaraDB for RDS instance, see "Configure a whitelist" in ApsaraDB for RDS User Guide.
3. Parameter settings used to connect to the cloud data source ApsaraDB RDS for MySQL are obtained.

Procedure

1. [Create a data source.](#)
2. Click the **MySQL** card and configure the parameters in the dialog box that appears.

Add a cloud data source ApsaraDB RDS for MySQL

Add MySQL Database [X]

* Name: Enter a database name to be displayed.

* Database Address: Enter a hostname.

* Port Number: 3306

* Database: Enter a database name.

* Username: Enter a username.

* Password: Enter the password.

VPC Data Source:

* AccessKey ID: Enter the AccessKey ID.

* AccessKey Secret: Enter the AccessKey Secret.

* Instance ID: Enter the instance ID.

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close Test Connection OK

Note If the data source is connected over a VPC, select **VPC Data Source** and configure the parameters.

Parameter	Description
Name	The name of the data source, which is user-defined.
Database Address	The hostname or IP address of the database that is used as the data source.
Port Number	The port that you use to access the database. Default value: 3306.
Database	The name of the database that you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database. Note If you do not know the username and password, contact the data warehouse administrator.

Parameter	Description
AccessKey ID	The AccessKey ID that you use to purchase the database instance.
AccessKey Secret	The AccessKey secret that you use to purchase the database instance.
Instance ID	The ID of the database instance.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

After a data source is added, you cannot add it again. If you attempt to add a data source for the second time, an error message appears.

7.4.2.2.4. Add an ApsaraDB RDS for SQL Server data source

This topic describes how to add an ApsaraDB RDS for SQL Server data source.

Prerequisites

1. An ApsaraDB RDS for SQL Server instance is created. For information about how to create an ApsaraDB for RDS instance, see "Create an instance" in ApsaraDB for RDS User Guide.
2. Available IP addresses of machines are obtained and added to a whitelist of the ApsaraDB for RDS instance. For information about how to query the IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add the IP addresses to a whitelist of the ApsaraDB for RDS instance, see "Configure a whitelist" in ApsaraDB for RDS User Guide.
3. Parameter settings used to connect to the cloud data source ApsaraDB RDS for SQL Server are obtained.

Context

The method of using an ApsaraDB RDS for SQL Server database as a data source is similar to that of using an ApsaraDB RDS for MySQL database as a data source. The only difference is that you must set the Schema parameter when you use an ApsaraDB RDS for SQL Server database as a data source.

Procedure

1. [Create a data source](#).
2. Click the **SQL Server** card and configure the parameters in the dialog box that appears.

Add SQL Server Database ✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

VPC Data Source: ⓘ

* AccessKey ID:

* AccessKey Secret:

* Instance ID:

ⓘ Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see [Quick BI User Guide > Create a data source > Data sources from cloud databases.](#)

ⓘ **Note** If the data source is connected over a VPC, select **VPC Data Source** and configure the parameters.

Parameter	Description
Name	The name of the data source, which is user-defined.
Database Address	The hostname or IP address of the database that is used as the data source.
Port Number	The port that you use to access the database. Default value: 1433.
Database	The name of the database that you want to access.
Schema	The database schema. Default value: dbo.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
AccessKey ID	The AccessKey ID that you use to purchase the database instance.
AccessKey Secret	The AccessKey secret that you use to purchase the database instance.
Instance ID	The ID of the database instance.

3. Click **Test Connection** to test connectivity with the database.

4. After the database passes the connectivity test, click **OK**.

7.4.2.2.5. Add an AnalyticDB for MySQL 2.0 data source

This topic describes how to add an AnalyticDB for MySQL 2.0 data source.

Prerequisites

An AnalyticDB database is created in the AnalyticDB console, and parameter settings used to connect to the AnalyticDB data source are obtained. For information about how to create an AnalyticDB database, see "Create a database" in AnalyticDB for MySQL User Guide.

Context

AnalyticDB for MySQL is formerly known as AnalyticDB.

Procedure

1. On the Data Sources page, click **Create Data Source** in the upper-right corner. For more information, see [Create a data source](#).
2. On the Add Data Source page, click the **Cloud Data Sources** tab. Then, click **AnalyticDB for MySQL 2.0** and set the parameters for connecting to the data source in the dialog box that appears.

The screenshot shows a dialog box titled "Add AnalyticDB for MySQL 2.0 Database". It contains the following fields and labels:

- Name:** Enter a database name to be displayed.
- Database Address:** Enter a hostname.
- Port Number:** 3306
- Database:** Enter a database name.
- AccessKey ID:** Enter the AccessKey ID.
- AccessKey Secret:** Enter the AccessKey Secret.

At the bottom of the dialog are three buttons: "Close", "Test Connection", and "Add".

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database.
Database	The name of the database to which you want to access.
AccessKey ID	The AccessKey ID used to identify a visitor.
AccessKey Secret	The AccessKey secret is used to encrypt the signature string on the client and decrypt the signature string on the server for authentication. Keep the AccessKey secret confidential.

3. Click **Test Connection** to test connectivity with the data source.

4. After the connection is established, click **Add**.

7.4.2.2.6. Add a HybridDB for MySQL data source

This topic describes how to add a HybridDB for MySQL data source.

Prerequisites

1. A HybridDB for MySQL instance is created.
2. The available machine IP addresses are obtained and added to a whitelist of the HybridDB for MySQL instance. For more information about how to query the available IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#).
3. Parameter settings used to connect to the HybridDB for MySQL data source are obtained.

Context

The procedure for adding a HybridDB for MySQL data source is similar to that for adding an SQL Server data source. The only difference is that the default port is the port specific to HybridDB for MySQL.

Procedure

1. [Create a data source](#).
2. Click **HybridDB for MySQL** and configure the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 3306.

Parameter	Description
Database	The name of the database to which you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.7. Add an AnalyticDB for PostgreSQL data source

This topic describes how to add an AnalyticDB for PostgreSQL data source.

Prerequisites

1. An AnalyticDB for PostgreSQL instance is created. For information about how to create an AnalyticDB for PostgreSQL instance, see "Create an instance" in AnalyticDB for PostgreSQL User Guide.
2. The available machine IP addresses are obtained and added to a whitelist of the AnalyticDB for PostgreSQL instance. For information about how to query the available IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add the IP addresses to a whitelist of the AnalyticDB for PostgreSQL instance, see "Configure a whitelist" in AnalyticDB for PostgreSQL User Guide.
3. Parameter settings used to connect to the AnalyticDB for PostgreSQL data source are obtained.

Context

The procedure for adding an AnalyticDB for PostgreSQL data source is similar to that for adding an SQL Server data source. The default port is the port specific to AnalyticDB for PostgreSQL.

Procedure

1. [Create a data source](#).
2. Click the **AnalyticDB for PostgreSQL** card and configure the parameters for connecting to the data source in the dialog box that appears.

Add AnalyticDB for PostgreSQL Database ✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

! Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see [Quick BI User Guide > Create a data source > Data sources from cloud databases.](#)

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port that you use to access the database. Default value: 5432.
Database	The name of the database to which you want to access.
Schema	The database schema. Default value: public.
Username	The username used to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.8. Add an AnalyticDB for PostgreSQL data source

This topic describes how to add an AnalyticDB for PostgreSQL data source.

Prerequisites

1. An RDS for PostgreSQL instance is created. For information about how to create an ApsaraDB for RDS instance, see "Create an instance" in ApsaraDB for RDS User Guide.
2. Available IP addresses of machines are obtained and added to a whitelist of the ApsaraDB for RDS instance. For information about how to query the IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add the IP addresses to a whitelist of the ApsaraDB for RDS instance, see "Configure a whitelist" in ApsaraDB for RDS User Guide.
3. Parameter settings used to connect to a cloud data source AnalyticDB for PostgreSQL are obtained.

Procedure

1. Create a data source.
2. Click **PostgreSQL** and configure the required parameters in the dialog box that appears.

Add PostgreSQL Database

* Name: Enter a database name to be displayed.

* Database Address: Enter a hostname or an IP address.

* Port Number: 5432

* Database: Enter a database name.

Schema: public

* Username: Enter a username.

* Password: Enter the password.

SSL:

① Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close Test Connection Add

Note After you select the **SSL** option, the data source supports the interactive query service MaxCompute Lightning provided by MaxCompute.

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port that you use to access the database. Default value: 5432.
Database	The name of the database that you want to access.
Schema	Default value: public.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.9. Add an ApsaraDB RDS for PPAS data source

This topic describes how to add an ApsaraDB RDS for PPAS data source.

Prerequisites

1. An RDS for PPAS instance is created. For information about how to create an ApsaraDB for RDS instance, see "Create an instance" in ApsaraDB for RDS User Guide.
2. The available IP addresses are obtained and added to a whitelist of the ApsaraDB for RDS instance. For information about how to query the IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add IP addresses to a whitelist of the ApsaraDB for RDS instance, see "Configure a whitelist" in ApsaraDB for RDS User Guide.
3. Parameter settings used to connect to the PPAS data source are obtained.

Procedure

1. On the Data Sources page, click Create Data Source in the upper-right corner. For more information, see [Create a data source](#).
2. Click the PPAS card and configure the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name of the data source that is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port that you use to access the database. Default value: 5432.
Database	The name of the database that you want to access.
Schema	The database schema. Default value: public.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.10. Add a cloud data source Data Lake Analytics

This topic describes how to add a cloud data source Data Lake Analytics (DLA).

Prerequisites

Parameter settings used to connect to the cloud data source DLA are obtained.

Procedure

1. [Create a data source](#).
2. Click the **Data Lake Analytics** card and configure the required parameters in the dialog box that appears.

Parameter	Description
Name	The name of the data source, which is user-defined.
Database Address	The hostname or IP address of the database that is used as the data source.
Port Number	The port that you use to access the database. Default value: 10000.
Database	The name of the database that you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **Add**.

7.4.2.2.11. Add a Hive data source

This topic describes how to add a Hive data source.

Prerequisites

Parameter settings used to connect to the Hive data source are obtained.

Procedure

1. [Create a data source](#).
2. Click **Hive** and configure the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 10000.
Database	The name of the database to which you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.12. Add a PolarDB-X data source

This topic describes how to add a PolarDB-X data source.

Prerequisites

- A PolarDB-X instance is created. For information about how to create a PolarDB-X instance, see "Create an instance" in PolarDB-X User Guide.
- Available IP addresses of machines are obtained and manually added to a whitelist of the PolarDB-X instance. For information about how to query the IP addresses, see [Add the IP addresses of a Quick BI cluster to a database whitelist](#). For information about how to add the IP addresses to a whitelist of the PolarDB-X instance, see "Configure a whitelist" in PolarDB-X User Guide.
- Parameter settings used to connect to the PolarDB-X data source are obtained.

Procedure

1. [Create a data source](#).
2. On the Add Data Source page, click the Cloud Data Sources tab. Then, click the **DRDS** card and configure the parameters for connecting to the data source in the dialog box that appears.

Add DRDS Database

* Name: Enter a database name to be displayed.

* Database Address: Enter a hostname or an IP address.

* Port Number: 3306

* Database: Enter a database name.

* Username: Enter a username.

* Password: Enter the password.

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close Test Connection Add

Parameter	Description
Name	The name of the data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port that you use to access the database. Default value: 3306.
Database	The name of the database that you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.13. Add a cloud Presto data source

This topic describes how to add a cloud Presto data source.

Prerequisites

Parameter settings used to connect to the Presto data source are obtained.

Procedure

1. On the Data Sources page, click **Create Data Source** in the upper-right corner. For more information, see [Create a data source](#).
2. On the Add Data Source page, click the **Cloud Data Sources** tab. Then, click **Presto** and set the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 3306.
catalog	The name of the data source that you want to use Presto to query.
Schema	The current logon username is used by default.
Username	The username used to access the database.
Password	The password used to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the connection is established, click **Add**.

7.4.2.2.14. Add an AnalyticDB for MySQL V3.0 data source

This topic describes how to add an AnalyticDB for MySQL V3.0 data source.

Prerequisites

An AnalyticDB for MySQL database is created in the AnalyticDB for MySQL console, and parameter settings for connecting to the database are obtained. For information about how to create an AnalyticDB for MySQL database, see "Create a database" in AnalyticDB for MySQL User Guide.

Context

AnalyticDB for MySQL is formerly known as AnalyticDB.

Procedure

1. **Create a data source.**
2. Click the **AnalyticDB for MySQL 3.0** card and configure the required parameters in the dialog box that appears.

The screenshot shows a dialog box titled "Add AnalyticDB for MySQL 3.0 Database". It contains the following fields and labels:

- * Name: Enter a database name to be displayed.
- * Database Address: Enter a hostname.
- * Port Number: 3306
- * Database: Enter a database name.
- * Account: Enter an account
- * Password: Enter the password.

At the bottom of the dialog are three buttons: "Close", "Test Connection", and "Add".

Parameter	Description
Name	The name of the data source that is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port that you use to access the database.
Database	The name of the database that you want to access.
Account	The username that you use to access the database.
Password	The password that you use to access the database.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.2.15. Add a PolarDB for MySQL data source

This topic describes how to add a PolarDB for MySQL data source.

Prerequisites

Parameter settings used to connect to the PolarDB for MySQL data source are obtained.

Procedure

1. **Create a data source.**
2. Click the **PolarDB for MySQL** card and configure the required parameters in the dialog box that appears.

Note If the data source is connected over a VPC, select VPC Data Source and configure the required parameters.

Add PolarDB for MySQL Database
✕

* Name:

* Database Address:

* Port Number:

* Database:

* Username:

* Password:

VPC Data Source:

* AccessKey ID:

* AccessKey Secret:

* Cluster ID:

* Region:

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close
Test Connection
Add

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 3306.
Database	The name of the database to which you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
AccessKey ID	The AccessKey ID that you use to purchase the instance where the database resides.
AccessKey Secret	The AccessKey Secret that you use to purchase the instance where the database resides.
Cluster ID	The ID of the cluster.
Region	The region where the instance resides.

Parameter	Description
-----------	-------------

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3. User-created data sources

7.4.2.3.1. Add a user-created MySQL data source

This topic describes how to add a user-created MySQL data source.

Prerequisites

Parameter settings used to connect to the MySQL database are obtained.

Context

The procedure for configuring a user-created MySQL data source is similar to that for configuring an ApsaraDB RDS for MySQL data source. You must perform the following operations to open the specific port of the firewall to allow Quick BI to access the MySQL database:

1. Run the following command to open the configuration file of the firewall:

```
vi /etc/sysconfig/iptables
```

2. Add the following command to the configuration file:

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT
```

3. Restart the iptable service after the configuration is complete.

```
service iptables restart
```

Procedure

1. Go to the Add Data Source page. For more information, see [Add a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click **MySQL** and configure the parameters for connecting to the database in the dialog box that appears.

Add PolarDB for MySQL Database ✕

* **Name:**

* **Database Address:**

* **Port Number:**

* **Database:**

* **Username:**

* **Password:**

VPC Data Source:

* **AccessKey ID:**

* **AccessKey Secret:**

* **Cluster ID:**

* **Region:** ▼

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close
Test Connection
Add

Note If you select `ssh`, you can access the database by using an SSH tunnel.

Parameter	Description
Name	The name of the data source.
Database Address	The hostname or IP address of the database that is used as the data source.
Port Number	The port that you use to access the database. Default value: 3306.
Database	The name of the database that you want to access.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username that you use to access the SSH host.
SSH Password	The password that you use to access the SSH host.
SSH Port Number	The port number that you use to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.2. Add a user-created SQL Server data source

This topic describes how to add a user-created SQL Server data source.

Prerequisites

Parameter settings used to connect to the SQL Server database are obtained.

Context

The procedure for configuring a user-created SQL Server data source is similar to that for configuring an ApsaraDB RDS for SQL Server data source. You must perform the following operations to open the specified port of the firewall to allow Quick BI to access the SQL Server database:

1. Run the following command to open the configuration file of the firewall:

```
vi /etc/sysconfig/iptables
```

2. Add the following command to the configuration file:

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT
```

3. After the configuration is complete, run the following command to restart the iptable service:

```
service iptables restart
```

Procedure

1. Go to the Add Data Source page. For more information, see [Add a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click the **SQL Server** card and configure the parameters for connecting to the database in the dialog box that appears.

Add SQL Server Database
✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

ssh:

SSH Host:

SSH Username:

SSH Password:

SSH Port Number:

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close
Test Connection
Add

Note If you select ssh, you can access the data source by using an SSH tunnel.

Parameter	Description
Name	The name of the data source.
Database Address	The hostname or IP address of the database.
Port	The port that you use to access the database. Default value: 1433.
Database	The name of the database that you want to access.
Schema	The database schema. Default value: dbo.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username that you use to access the SSH host.
SSH Password	The password that you use to access the SSH host.

Parameter	Description
SSH Port Number	The port number that you use to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **Add**.

 **Note** You cannot add a data source that has been added. If you attempt to add it again, an error message appears.

7.4.2.3.3. Add a user-created PostgreSQL data source

This topic describes how to add a user-created PostgreSQL data source.

Prerequisites

Parameter settings used to connect to the PostgreSQL database are obtained.

Context

The procedure for configuring a user-created PostgreSQL data source is similar to that for configuring an ApsaraDB RDS for PostgreSQL data source. You must perform the following operations to open the specified port of the firewall to allow Quick BI to access the PostgreSQL database:

1. Run the following command to open the configuration file of the firewall:

```
vi /etc/sysconfig/iptables
```

2. Add the following command to the configuration file:

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 5432 -j ACCEPT
```

3. After the configuration is complete, run the following command to restart the iptables service:

```
service iptables restart
```

Procedure

1. [Create a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click the **PostgreSQL** card and configure the parameters for connecting to the database in the dialog box that appears.

Add PostgreSQL Database
✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

User-created ECS Data Source (VPC):

* AccessKey ID:

* AccessKey Secret:

* Instance ID:

* ECS Instance Region:

SSL:

ssh:

SSH Host:

SSH Username:

SSH Password:

SSH Port Number:

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see [Quick BI User Guide > Create a data source > Data sources from cloud databases.](#)

Close
Test Connection
Add

Note If the data source is connected over a VPC, select **User-created ECS Data Source (VPC)** and configure the parameters. If you select **SSL**, MaxCompute Lightning is supported. If you select **ssh**, you can access the data source by using an SSH tunnel.

Parameter	Description
Name	The name of the data source.
Database Address	The IP address of the database.
Port Number	The port that you use to access the database. Default value: 5432.
Database	The name of the database that you want to access.
Schema	The database schema. Default value: public.
Username	The username that you use to access the database.

Parameter	Description
Password	The password that you use to access the database.
AccessKey ID	The AccessKey ID that you use to purchase the instance where the database resides.
AccessKey Secret	The AccessKey secret that you use to purchase the instance where the database resides.
Instance ID	The ID of the instance.
ECS Instance Region	The region where the data source is located.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username that you use to access the SSH host.
SSH Password	The password that you use to access the SSH host.
SSH Port Number	The port number that you use to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.4. Add a user-created Oracle data source

This topic describes how to add a user-created Oracle data source.

Prerequisites

Parameter settings used to connect to the Oracle database are obtained.

Procedure

1. [Add a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click the **Oracle** card and configure the parameters for connecting to the database in the dialog box that appears.

Add Oracle Database
✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

ssh:

SSH Host:

SSH Username:

SSH Password:

SSH Port Number:

Close
Test Connection
Add

? **Note** If you select `ssh`, you can access the data source by using an SSH tunnel.

Parameter	Description
Name	The name to be displayed for the connected data source.
Database Address	Set the parameter to IP address.
Port Number	The port number of the database. The default port is 1521.
Database	The name of the database to which you want to access.
Schema	The schema of the database. The default value is public.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username that you use to access the SSH host.
SSH Password	The password that you use to access the SSH host.
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.5. Add a self-managed Hive data source

This topic describes how to add a self-managed Hive data source.

Prerequisites

Parameter settings for connecting to the Hive data source are obtained.

Procedure

1. [Create a data source.](#)
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click **Hive** and set the parameters for connecting to the data source in the dialog box that appears.

The screenshot shows a dialog box titled "Add Hive Database" with a close button (X) in the top right corner. The dialog contains the following fields and values:

- Name:** * Name: Enter a database name to be displayed. (Required)
- Database Address:** * Database Address: IP
- Port Number:** * Port Number: 10000
- Database:** * Database: Enter a database name. (Required)
- Username:** * Username: Enter a username. (Required)
- Password:** * Password: Enter the password. (Required)
- ssh:** ssh:
- SSH Host:** SSH Host: Enter a hostname or an IP address.
- SSH Username:** SSH Username: Enter a username.
- SSH Password:** SSH Password: Enter the password.
- SSH Port Number:** SSH Port Number: 22

At the bottom of the dialog, there are three buttons: "Close", "Test Connection", and "Add".

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	Set the parameter to IP address.
Port Number	The port number of the database. The default port is 10000.
Database	The name of the database to which you want to access.
Username	The username used to access the database.
Password	The password used to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username used to access the SSH host.

Parameter	Description
SSH Password	The password used to access the SSH host.
SSH Port Number	The port number that you use to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.6. Add a self-managed Vertica data source

This topic describes how to add a self-managed Vertica data source.

Prerequisites

Parameter settings for connecting to the Vertica data source are obtained.

Procedure

1. [Create Data Source](#).
2. Click the **Vertica** icon and set the parameters for connecting to the data source in the dialog box that appears.

The screenshot shows a dialog box titled "Add Vertica Database" with a close button (X) in the top right corner. The dialog contains the following fields and values:

- Name:** * Name: Enter a database name to be displayed.
- Database Address:** * Database Address: IP
- Port Number:** * Port Number: 5433
- Database:** * Database: Enter a database name.
- Schema:** Schema: public
- Username:** * Username: Enter a username.
- Password:** * Password: Enter the password.
- ssh:** ssh:
- SSH Host:** SSH Host: Enter a hostname or an IP address.
- SSH Username:** SSH Username: Enter a username.
- SSH Password:** SSH Password: Enter the password.
- SSH Port Number:** SSH Port Number: 22

At the bottom of the dialog, there are three buttons: "Close", "Test Connection", and "Add".

Parameter	Description
Name	The name to be displayed for the connected data source.
Database Address	The hostname or IP address of the database.

Parameter	Description
Port Number	The port number of the database. The default port is 5433.
Database	The name of the database to which you want to access.
Schema	The schema of the database. The default value is public.
Username	The username used to access the database.
Password	The password used to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username used to access the SSH host.
SSH Password	The password used to access the SSH host.
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.7. Add a user-created IBM DB2 LUW data source

This topic describes how to add a user-created IBM DB2 LUW data source.

Prerequisites

Parameter settings for connecting to the IBM DB2 LUW data source are obtained.

Procedure

1. [Create a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click **IBM DB2 LUW** and set the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name to be displayed for the connected data source.
Database Address	The IP address of the database.
Port Number	The port number of the database. The default port is 50000.
Database	The name of the database that you want to access.
Schema	The schema of the database. The default value is DB2INST1.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username that you use to access the SSH host.
SSH Password	The password that you use to access the SSH host.
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.8. Add a user-created SAP IQ (Sybase IQ) data source

This topic describes how to add a user-created SAP IQ (Sybase IQ) data source.

Prerequisites

Parameter settings for connecting to the SAP IQ (Sybase IQ) data source are obtained.

Procedure

1. On the Data Sources page, click Create Data Source in the upper-right corner. For more information, see [Create a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click **SAP IQ (Sybase IQ)** and set the parameters for connecting to the data source in the dialog box that appears.

Parameter	Description
Name	The name to be displayed for the connected data source.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 2638.
Database	The name of the database that you want to access.
Schema	The schema of the database. The default value is sybase.
Username	The username used to access the database.
Password	The password used to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username used to access the SSH host.
SSH Password	The password used to access the SSH host.

Parameter	Description
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the data source.
4. After the connection is established, click **Add**.

7.4.2.3.9. Add a user-created SAP HANA data source

This topic describes how to add a user-created SAP HANA data source.

Prerequisites

Parameter settings used to connect to the SAP HANA database are obtained.

Procedure

1. [Create a data source](#).
2. On the Add Data Source page, click the User-created Data Sources tab. Then, click the **SAP HANA** card and configure the parameters for connecting to the data source in the dialog box that appears.

Add SAP HANA Database
✕

* Name:

* Database Address:

* Port Number:

* Database:

Schema:

* Username:

* Password:

ssh:

SSH Host:

SSH Username:

SSH Password:

SSH Port Number:

Close
Test Connection
Add

Note If you select **ssh**, you can access the database by using an SSH tunnel.

Parameter	Description
Name	The name to be displayed for the connected data source.
Database Address	The hostname or IP address of the database.

Parameter	Description
Port Number	The port that you use to access the database. Default value: 30015.
Database	The name of the database that you want to access.
Schema	The database schema. Default value: public.
Username	The username that you use to access the database.
Password	The password that you use to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username used to access the SSH host.
SSH Password	The password used to access the SSH host.
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the database.
4. After the database passes the connectivity test, click **OK**.

7.4.2.3.10. Add a user-created Presto data source

This topic describes how to add a user-created Presto data source.

Prerequisites

Parameter settings for connecting to the Presto data source are obtained.

Procedure

1. On the Data Sources page, click **Create Data Source** in the upper-right corner. For more information, see [Create a data source](#).
2. On the Add Data Source page, click the **User-created Data Sources** tab. Then, click **Presto** and set the parameters for connecting to the data source in the dialog box that appears.

Add Presto Database
✕

* Name:

* Database Address:

* Port Number:

* catalog:

Schema:

* Username:

* Password:

ssh:

SSH Host:

SSH Username:

SSH Password:

SSH Port Number:

Note: To add IP addresses to the whitelist, log on to the RDS console and perform required operations. For more information about how to view the whitelisted IP addresses, see Quick BI User Guide > Create a data source > Data sources from cloud databases.

Close
Test Connection
Add

Note If you select `ssh`, you can access the data source by using an SSH tunnel.

Parameter	Description
Name	The name to be displayed for the connected data source, which is user-defined.
Database Address	The hostname or IP address of the database.
Port Number	The port number of the database. The default port is 3306.
catalog	The name of the data source that you want to use Presto to query.
Schema	The current logon username is used by default.
Username	The username used to access the database.
Password	The password used to access the database.
SSH Host	The hostname or IP address of the SSH host.
SSH Username	The username used to access the SSH host.
SSH Password	The password used to access the SSH host.

Parameter	Description
SSH Port Number	The port number used to access the SSH host. Set the value to 22.

3. Click **Test Connection** to test connectivity with the data source.
4. After the connection is established, click **Add**.

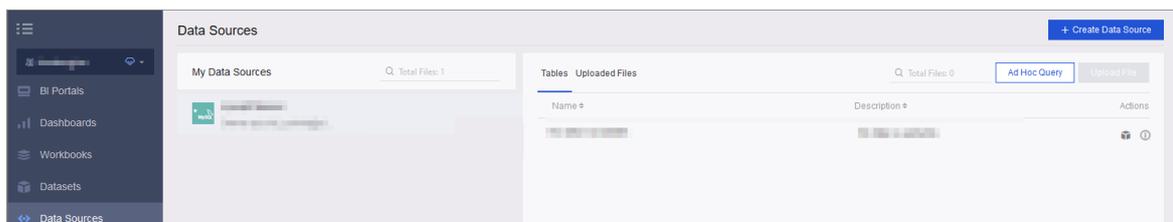
7.4.2.4. Go to the Data Sources page

This topic describes how to go to the Data Sources page.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.

On this page, you can manage data sources. For example, you can **create**, **edit**, **replace**, and **delete** a data source.



7.4.2.5. Create a data source

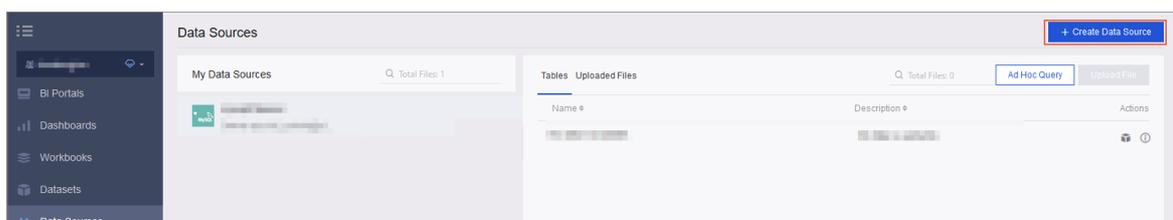
This topic describes how to create a data source.

Prerequisites

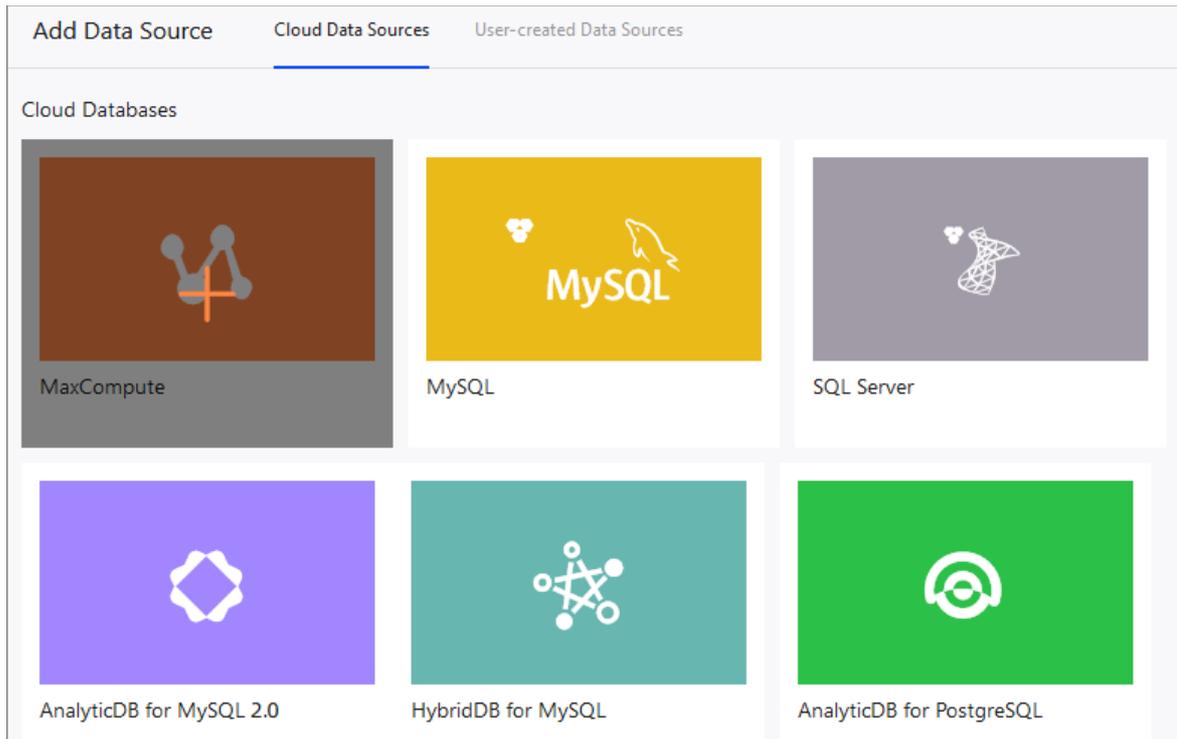
The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane of the workspace page, click **Data Sources**.
4. In the upper-right corner of the Data Sources page, click **Create Data Source**.



5. In the **Add Data Source** dialog box, select a data source.



- If you want to add a cloud data source, click the **Cloud Data Sources** tab.
 - If you want to add a user-created data source, click the **User-created Data Sources** tab.
6. Select the data source type. In the dialog box that appears, enter the information required for connecting to the data source, and click **OK**.

7.4.2.6. Edit a data source

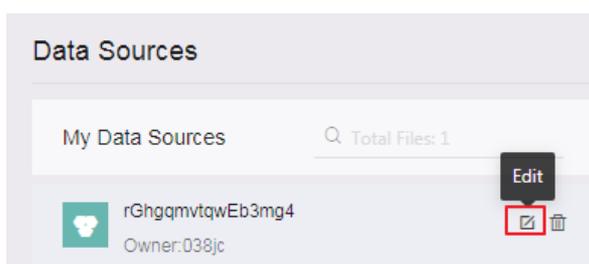
This topic describes how to edit a data source.

Prerequisites

- The Quick BI service is purchased.
- A data source is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. Find the data source that you want to edit in the data source list.
5. Click the  icon and edit the data source.



7.4.2.7. Replace a data source

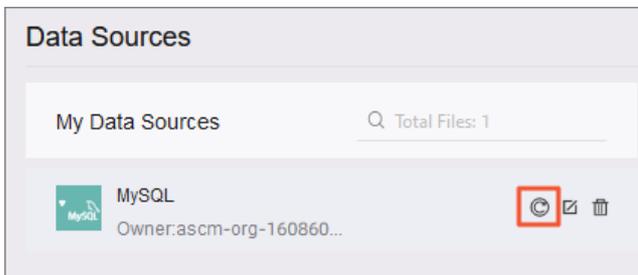
This topic describes how to replace a data source.

Prerequisites

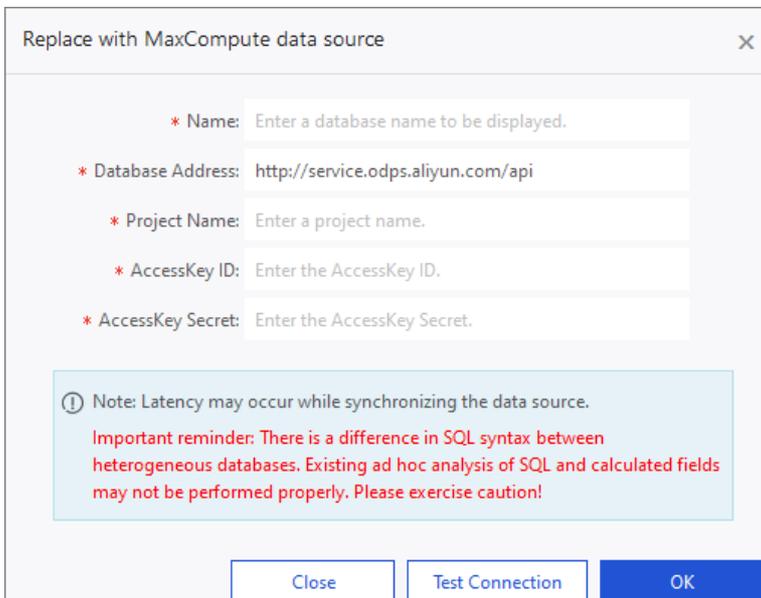
- The Quick BI service is purchased.
- A data source is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. Find the data source that you want to replace in the data source list.
5. Click the  icon and replace the data source with another one.



6. In the **Replace Data Source** dialog box, click the target data source and configure the parameters in the dialog box that appears.



7. After the parameters are configured, click **Test Connection**.
8. After the data source passes the connectivity test, click **OK**.

7.4.2.8. Delete a data source

This topic describes how to delete a data source.

Prerequisites

- The Quick BI service is purchased.
- A data source is created.

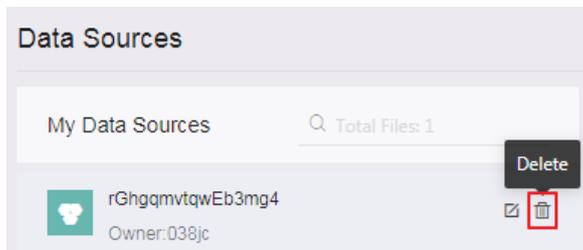
Context

If you have created datasets based on a data source, you cannot delete the data source.

✘ Cannot delete this data source; it has associated datasets.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. Find the data source that you want to delete in the data source list.
5. Click the  icon to delete the data source.



7.4.2.9. Search for a data source

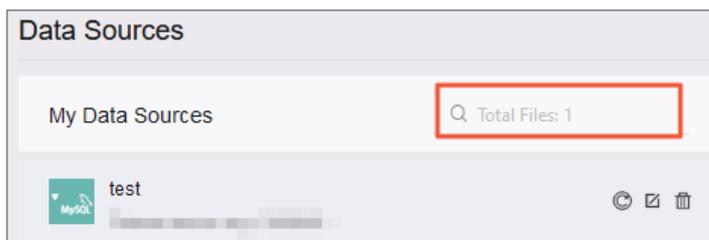
This topic describes how to search for a data source.

Prerequisites

- The Quick BI service is purchased.
- A data source is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. On the Data Sources page, enter a keyword of the data source that you want to search for in the search box.



5. Click the  icon.

7.4.2.10. Search for a table under a data source

This topic describes how to search for a specific table under a data source.

Prerequisites

The Quick BI service is purchased.

Procedure

1. Log on to the Quick BI console.
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. Select the target data source. All tables under the data source are listed in the right-side part of the page.
5. Enter a keyword of the table that you want to search in the search box.



6. Click the  icon.

7.4.2.11. View the details of a table under a data source

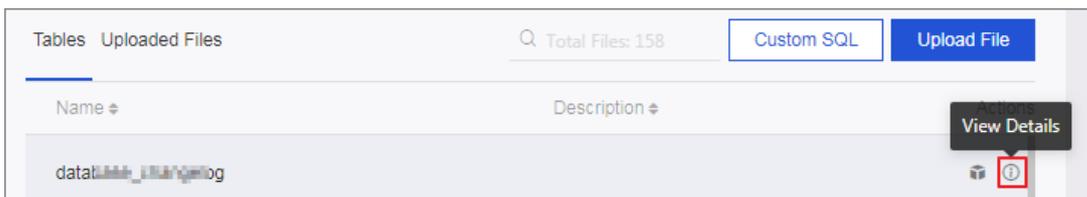
Quick BI allows you to view the tables and table details under a data source.

Prerequisites

The Quick BI service is purchased.

Procedure

1. Log on to the Quick BI console.
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. Select the target data source. All tables under the data source are listed in the right-side part of the page.
5. Find the table whose details you want to view and click the  icon in the Actions column.



7.4.3. Datasets

7.4.3.1. Overview of datasets

You can use tables from data sources to create datasets. The following topics describe common operations on datasets, for example, create, edit, and query a dataset.

You can use one of the following three methods to create a dataset:

- Create a dataset from a data source.
- Create a dataset by uploading a file.
- Create a dataset by using an SQL statement for ad hoc query.

7.4.3.2. Create datasets

7.4.3.2.1. Create a dataset based on a data source

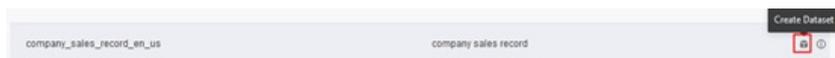
This topic describes how to create a dataset based on a data source.

Prerequisites

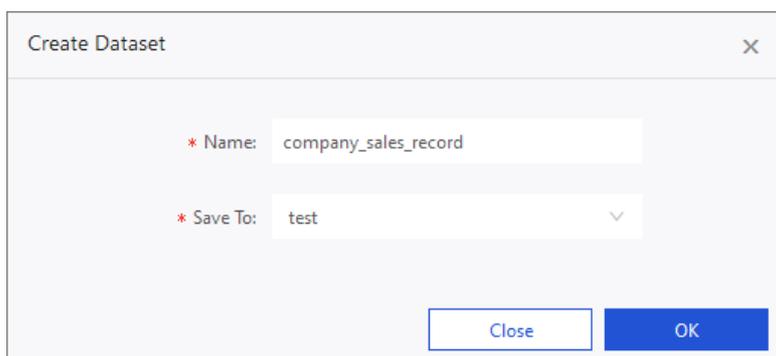
The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. On the Data Sources page, click the required data source. All tables under the data source are listed in the right part of the page.
5. Find the required table and click the  icon in the Actions column.



6. In the Create Dataset dialog box, enter a dataset name and location, and click **OK**.

A screenshot of the 'Create Dataset' dialog box. The dialog has a title bar with 'Create Dataset' and a close button. It contains two required fields: '* Name:' with the value 'company_sales_record' and '* Save To:' with a dropdown menu set to 'test'. At the bottom, there are two buttons: 'Close' and 'OK'.

After the dataset is created, the **Datasets** page appears. The new dataset is displayed in the dataset list and marked with NEW.



7.4.3.2.2. Create a dataset by uploading a file

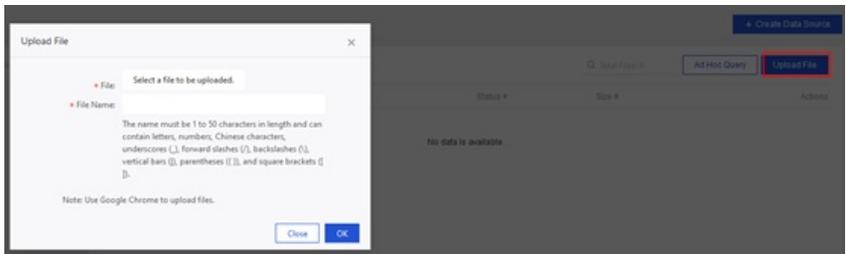
This topic describes how to create a dataset by uploading a file.

Prerequisites

- The Quick BI service is purchased.
- A data source is created.

Procedure

1. Log on to the Quick BI console.
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Data Sources**.
4. On the Data Sources page, click a data source, and then click **Upload File** in the upper-right corner of the page.
5. In the **Upload File** dialog box, select the target file, enter a file name, and then click **OK**.



Note After the file is uploaded, the Uploaded Files page appears.

6. On the Uploaded Files page, find the file you uploaded and click the  icon in the Actions column to create a dataset.

7.4.3.2.3. Create a dataset by using an SQL statement for ad hoc queries

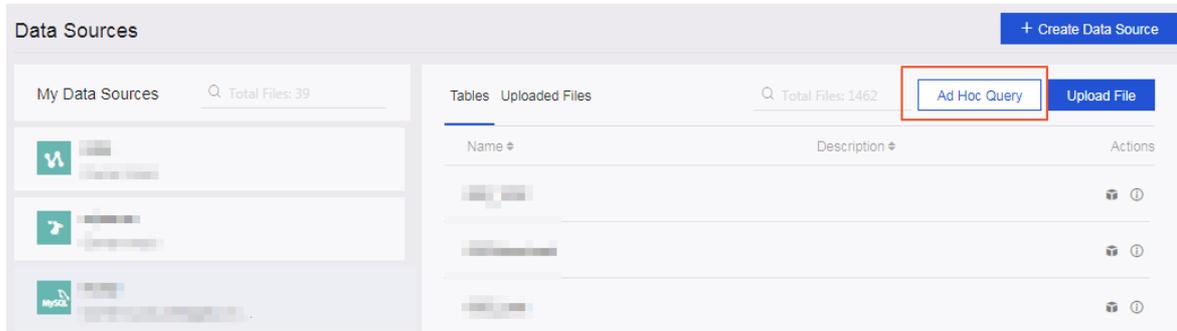
This topic describes how to create a dataset by using an SQL statement for ad hoc query to implement some complex logic for data modeling. Ad hoc queries support dynamic parameter passing to SQL statements. Modeling analysis based on dynamic parameter passing to SQL statements increases the depth of scenarios supported by Quick BI. This meets the requirements of complex data analysis scenarios.

Prerequisites

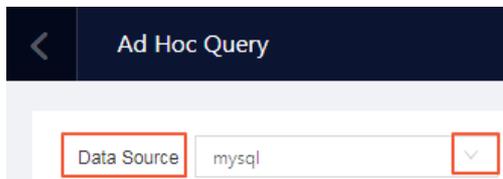
The Quick BI service is purchased.

Procedure

1. Log on to the Quick BI console.
2. Click the **Workspace** tab.
3. In the left-side navigation pane of the Workspace page, click **Data Sources**.
4. In the upper-right corner of the **Data Sources** page, click **Ad Hoc Query**.



5. On the **Ad Hoc Query** page, specify a data source.



6. Enter an SQL statement.

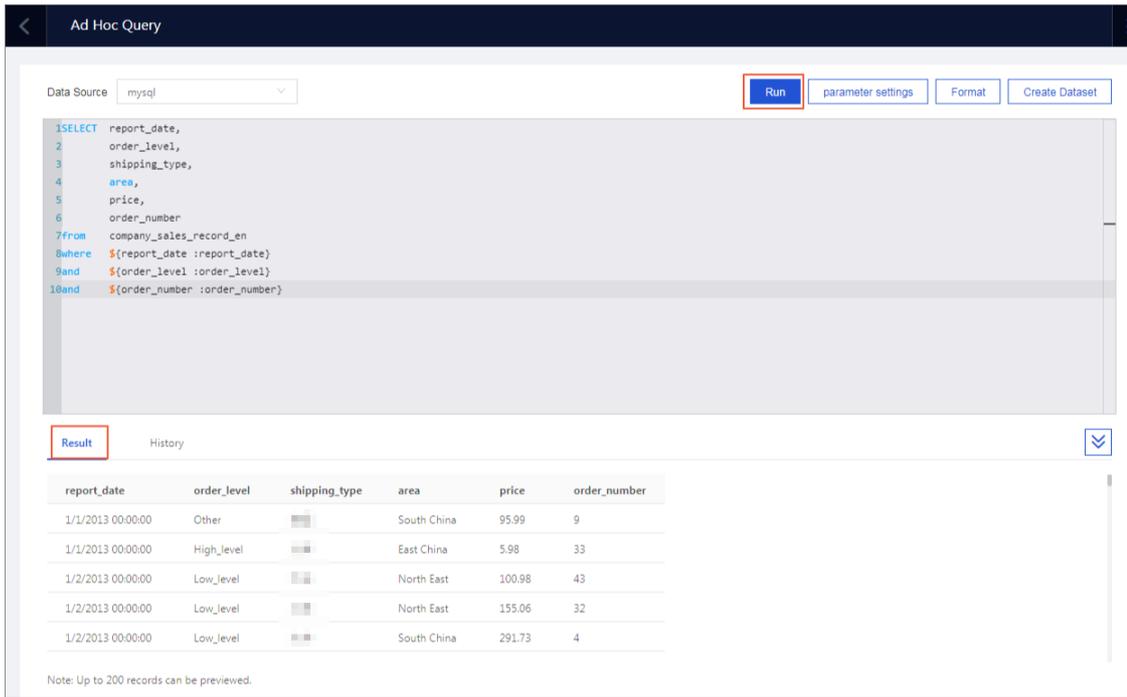
Sample SQL statement :

```
SELECT report_date,
       order_level,
       shipping_type,
       area,
       price,
       order_number
from company_sales_record
where ${report_date:report_date}
and  ${order_level:order_level}
and  ${order_number:order_number}
```

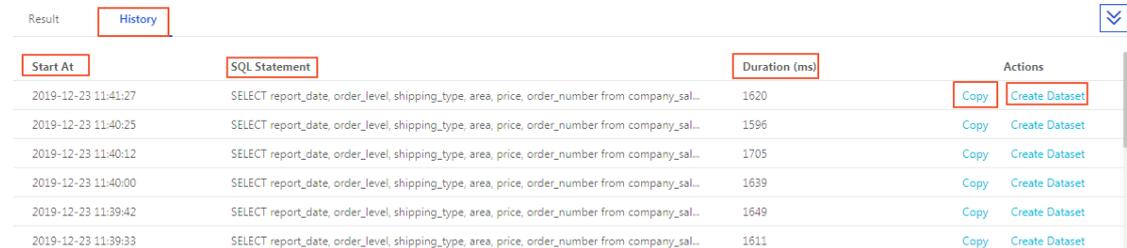
7. Click **Run** to execute the SQL statement.

You can view the execution results on the **Result** tab.

i. Click the **Result** tab.



ii. On the **History** tab, view the execution time, SQL statement, and time consumed by the ad hoc query.



- Click **Copy** in the Actions column to copy the SQL statement to the SQL statement input box.
- Click **Create Dataset**. A dataset is created by using the historical SQL statement.
- Click the  icon to hide the execution result.

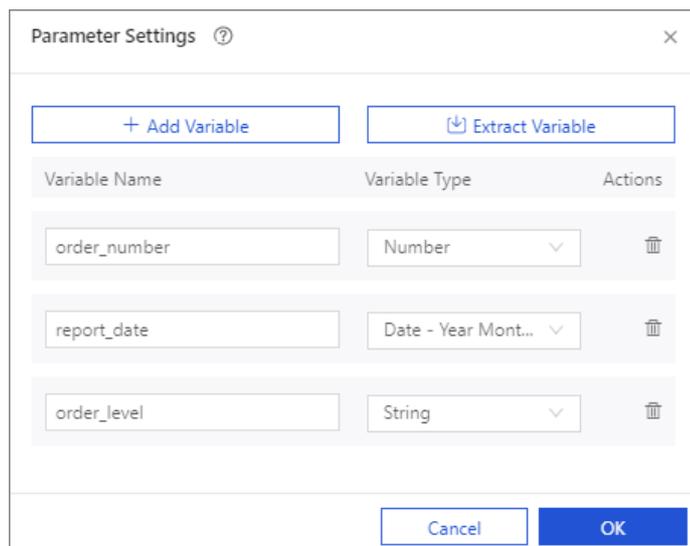
SQL statements for ad hoc queries support dynamic parameter passing and placeholders. When you create an SQL model, you can append SQL parameters to the WHERE clause in the format of \$ {Physical field name:Parameter alias}. The parameters can be referenced by the filter bar widget.

 **Note** Parameter fields are not displayed in the dataset, but are displayed in the filter bar widget.

Example of SQL statements:

```
SELECT report_date,
       order_level,
       shipping_type,
       area,
       price,
       order_number
from company_sales_record
where ${report_date:report_date}
and  ${order_level:order_level}
and  ${order_number:order_number}
```

8. Before you execute SQL statements, you can set and format parameters. After the SQL statements are executed, you can create datasets.
 - i. Click **parameter settings**. In the dialog box that appears, configure the parameters. You can add variables and modify variable types. The following variable types are supported: String, Number, Date - Year Month Day, Date - Year Month, Date - Year and Week, Date - Year, and Date - YYYYMMDD HH:MM:SS.
 - Click **New parameter variables** in the parameter settings dialog box, and specify **variable name** and **Variable type**. The variable name must be added to the WHERE clause of an SQL statement in the \${Physical field name:Variable name} format.
 - Click **Quick extraction** to obtain the variable aliases in the SQL statement. The default variable type is String, which can be changed.



After you configure the parameters, click **OK**.

- ii. Click **Format** to format SQL statements.
- iii. Click **Create Dataset** to create a dataset based on the query results of the current SQL statement.

For information about how to use the parameters in the filter bar widget, see [Use SQL statement parameters in the Filter Bar widget](#).

7.4.3.3. Specify a method to name dimensions and measures

Quick BI creates datasets based on the metadata of physical tables and converts fields in the physical tables to dimensions or measures in the datasets. The names of dimensions and measures are automatically generated. They can be the names or descriptions of the fields in the physical tables.

Prerequisites

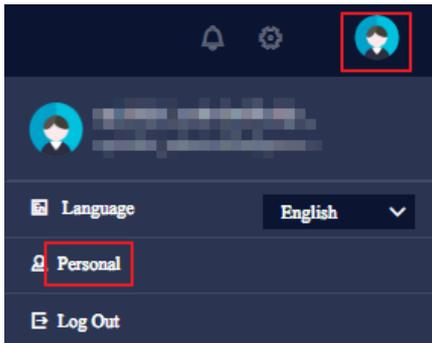
The Quick BI service is purchased.

Context

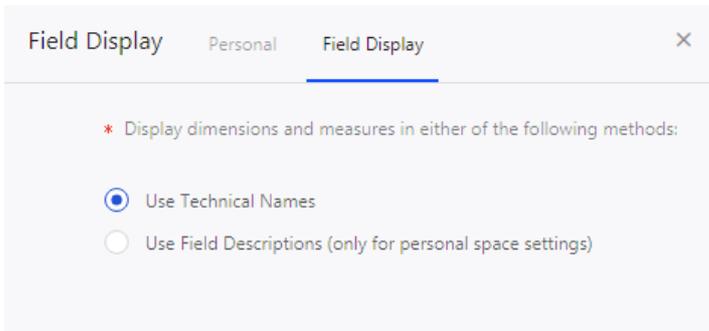
You can move the pointer over the profile picture and specify a method to name dimensions and measures in datasets. Dimensions and measures in datasets are named based on your setting.

Procedure

1. [Log on to the Quick BI console.](#)
2. Move the pointer over the profile picture in the upper-right corner and select **Personal**.



3. In the **Personal** dialog box, click the **Display** tab and select a method to name dimensions and measures in datasets.



7.4.3.4. Edit a dataset

7.4.3.4.1. Edit a dimension

If the data type of a field in a dataset table is String or Date/Time, the field is classified as a dimension by the system. You can edit the field in the Dimensions list.

Prerequisites

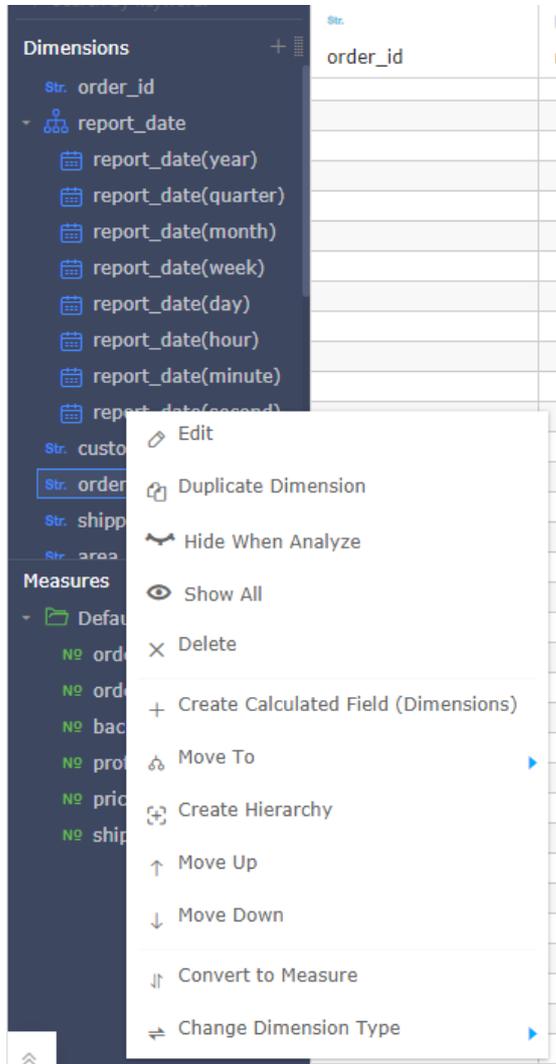
The Quick BI service is purchased.

A dataset is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.

4. On the Datasets page, find the required dataset and click the  icon in the Actions column to go to the dataset edit page.
5. Find the required field in the Dimensions list.
6. Right-click the field and select the operation you want to perform.



- **Edit**: Edit the name and description of the dimension.
When you edit a week dimension, you can configure the first day of a week.
- **Duplicate Dimension**: Duplicate the dimension. The name of the duplicate dimension ends with **Duplicate**.
- **Hide When Analyze**: Hide the dimension as required.
- **Show All**: Show all dimensions.
- **Delete**: Delete the dimension.
- **Create Calculated Field (Dimensions)**: Create a dimension and customize its calculation method.
- **Move To**: Move the dimension to an existing hierarchy for drilling.
- **Create Hierarchy**: Add the dimension field to a new hierarchy.
- **Move Up/Move Down**: Move the dimension. You can drag or right-click the dimension to move it.
- **Convert to Measure**: Convert the dimension to a measure.

- **Change Dimension Type:** Switch the dimension type among Date/Time (Source Format), Geo, String, and Number.

For example, you must change the data type of the province and city dimensions to Geo when you create a bubble map or colored map. Otherwise, the map cannot be created.

 **Note** You can duplicate, hide, and delete dimensions at different hierarchy levels.

7.4.3.4.2. Edit a measure

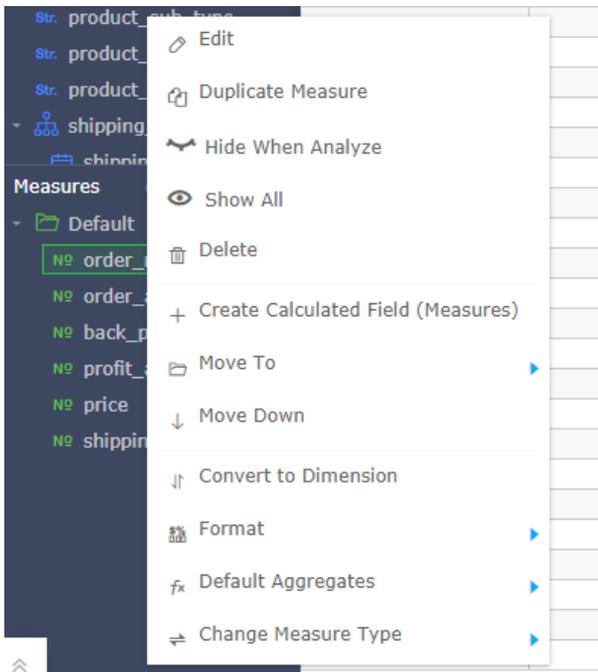
If the data type of a field in a dataset table is NUMBER, the field is classified as a measure by default. You can edit the field in the Measures list.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the Datasets page, click the required dataset to go to the dataset edit page.
5. Find the field that you want to edit in the Measures list.
6. Right-click the field and select the operation that you want to perform on the field as required.



- **Edit:** Edit the name and description of the measure.
- **Duplicate Measure:** Duplicate the measure. The name of the duplicate measure ends with **Duplicate**.
- **Hide When Analyze:** Hide the measure as required.
- **Show All:** Show all measures.
- **Delete:** Delete the measure.

- Create Calculated Field (Measures): Create a measure and customize its calculation method.
- Move To: Move the measure to an existing folder.
- Move Up/Move Down: Move the measure. You can drag or right-click the measure to move it.
- Convert to Dimension: Convert the measure to a dimension.
- Format: Specify the number format.
- Default Aggregates: Specify the aggregate function, such as SUM, MAX, or MIN.
- Change Measure Type: Switch the measure type between String and Number.

7.4.3.4.3. Toolbar and shortcut menu

On the dataset edit page, Quick BI provides a toolbar and a shortcut menu for you to edit a dataset and create reports.

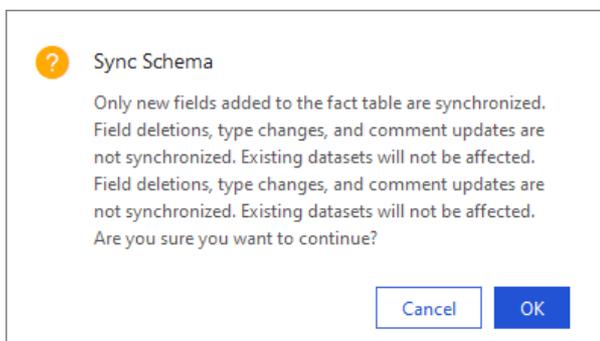
Toolbar



- Lock: provides a lock mechanism. Multiple users can edit a dataset at the same time. If you want to modify a dataset, you must first lock the dataset and refresh the page to update the dataset. Then, make and save the modifications. If you modify the dataset without refreshing the page, the changes made by other users are overwritten.
- Sync Schema: synchronizes the new fields in an online physical table to the dataset that is created based on the physical table.

However, if a field is modified or deleted from the physical table, the field in the dataset remains unchanged.

Sync Schema



- Refresh Preview: refreshes the dataset for preview. If you want to view the latest data, save the dataset and click Refresh Preview.
- Set Filter: filters out unnecessary data in the dataset to prevent full table scan.
- Save As: saves the current dataset as a new dataset. This function allows you to duplicate or backup a dataset.
- Save: saves the dataset.

After you add new fields, delete fields, or convert between dimensions and measures in a dataset, you must save the dataset. Then, you can refresh the page to view the updated dataset.

Shortcut menu

On the dataset edit page, click the  icon in the lower-left corner to expand the shortcut menu.

Short cut menu

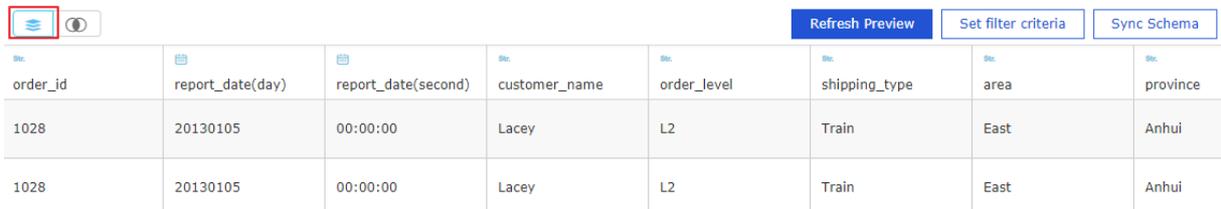


- Dashboards: You can click **Dashboards** to open the **Create Dashboard** dialog box.
- Workbooks: You can click **Workbooks** to go to the page for creating and editing a workbook.
- Datasets: You can click **Datasets** to go to the page for creating a dataset.
- BI Portals: You can click **BI Portals** to go to the page for creating and editing a BI portal.
- Retrieve Data: You can click **Retrieve Data** to go to the page for creating a data source.

7.4.3.4.4. Preview data

This topic describes how to preview data on the dataset edit page.

Click the  icon to preview data.



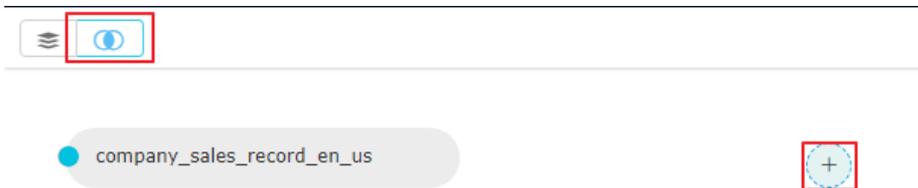
order_id	report_date(day)	report_date(second)	customer_name	order_level	shipping_type	area	province
1028	20130105	00:00:00	Lacey	L2	Train	East	Anhui
1028	20130105	00:00:00	Lacey	L2	Train	East	Anhui

7.4.3.4.5. Table join and examples

This topic describes how to join tables when you edit a dataset.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane of the Workspace page, click **Datasets**.
4. On the Datasets page, click a dataset that you want to manage and go to the dataset edit page.
5. Click the **Join** icon () to switch to the Table Join mode.



6. Click the  icon. The Build Table Join for company_sales_record Associated Mode dialog box appears. Quick BI supports five-layer horizontal join, the following figure shows details.



7. Select the fields that are used to join tables and specify a join type.

Build a table join model

Build Table Join for company_sales_record_en_us Associated Model ✕

Dataset Field	Join Type	Join Table	Join Field	Actions
order_id		company_sales_rec...	order_id	

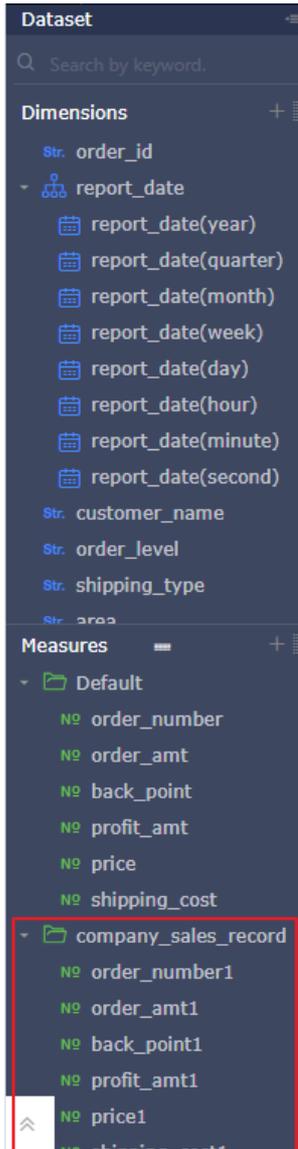
Add Join Field

Quick BI supports the following join types:

- Inner Join (): returns records with the same join fields in two tables.
- Left Outer Join (): returns all records in the left table and records in the right table that have the same join fields as the left table.
- All Join (): returns all records in two tables.

Note MySQL data sources do not support the All Join type.

8. Click **Join Field** to add multiple join fields.
9. Click **OK** to save the model.
10. Click **Save**.
11. After the model is saved, click the preview icon () to switch to the preview mode, and then click **Refresh Preview** to view the join results of the tables.



7.4.3.4.6. Calculated fields

7.4.3.4.6.1. Overview

Quick BI allows you to create fields from existing fields in data sources by using SQL functions. The new fields are referred to as calculated fields.

When you create a calculated field, take note of the following points:

- Fields that are created from the Dimensions list are classified as calculated dimensions. Fields that are created from the Measures list are classified as calculated measures.
- In the Expression field, you can select any functions or expressions supported by the current data source.
- You must enter the function name. The field name must be enclosed in a pair of brackets. You can enter the entire field name, or enter a left bracket ([) and select the required field from the displayed field list. You can also double-click the target field in the Dimensions or Measures list to add the field to the Expression field. Correct SQL expressions are highlighted.
- When you specify an SQL expression for a calculated field, do not use full-width punctuation marks. If an error occurs, check whether you have entered full-width punctuation marks in the expression field.

- After you create the calculated field, save the dataset and then refresh it to show the calculated field.
- You cannot use a calculated field in the expression of another calculated field. If a field is deleted from the physical table, a calculated field whose expression uses the deleted field becomes invalid.

7.4.3.4.6.2. Rules for using calculated fields

This topic describes the rules for using calculated fields.

- Calculated fields not using aggregate functions can be used as dimensions, or as measures after you specify the aggregate method. Calculated fields using aggregate functions can only be used as measures, and cannot be converted to dimensions.
- You can set the data type for a calculated field. Currently, you can set the data type of dimension fields to Number, Text, or Date/Time, and set the data type of measure fields to Number or Text.
- Similar to dimensions and measures generated by the original fields in a data source, dimensions and measures generated by calculated fields can be used in rows, columns, and filters, and can be selected on the Data tab for charts and maps. You can also convert a calculated field from dimension to measure or from measure to dimension.

 **Note** If you use the SUM or AVG aggregate function in an expression of a calculated field whose data type is Text and values are text data, an error occurs due to a failure in data type conversion.

7.4.3.4.6.3. Types of calculated measures

A calculated field in the measure category is a calculated measure. Calculated measures are classified into common measures and aggregate measures based on the expression type.

The following table lists differences between common measures and aggregate measures.

Common measure	Aggregate measure
Expressions exclude aggregate functions.	Expressions include aggregate functions.
You can change the aggregation method.	You cannot change the aggregation method.
You can convert common measures to dimensions.	You cannot convert aggregate measures to dimensions.
No aggregate functions are supported.	Supported aggregate functions: SUM, AVG, MIN, MAX, COUNT, and COUNT DISTINCT.

You can use the COUNT or COUNT DISTINCT function that has dimensions as its parameters to form a deduplicated aggregate measure.

For example, to calculate the average purchase price per user, you can use the following expression: `sum(purchase price)/countd(user ID)`. To calculate the proportion of order costs to order prices, you can use `sum(order cost)/sum(order price)` but not `avg(order cost/order price)`.

 **Note** Common measures cannot be used with aggregate measures. For example, `sum(order cost)/order price` is incorrect.

7.4.3.4.6.4. Expressions of calculated fields

This topic describes functions and arithmetic operations of calculated fields.

Aggregation methods

- To calculate the total order price: `sum([order_amt])`
- To calculate the average order price: `avg([order_amt])`
- To calculate the maximum order price: `max([order_amt])`
- To calculate the minimum order price: `min([order_amt])`
- To count the number of customers: `count([customer_name])`
- To count the number of unique customers: `count(distinct [customer_name])`

Basic operations

`order_cost = ([order_amt] - [profit_amt])/100`

Substring

`Substring([customer_name],1,1)`

Group values of a measure by using the CASE WHEN statement

- Group orders based on the order price
`CASE WHEN [order_amt] < 500 THEN 'small order' WHEN [order_amt] >= 500 AND [order_amt] < 2000 THEN 'medium order' WHEN [order_amt] >= 2000 AND [order_amt] < 5000 THEN 'big order' ELSE 'large order' END`
- Group dimension members by using the CASE WHEN statement. In this example, provinces are classified into a specific physical region.
`CASE WHEN [province] in ('Heilongjiang', 'Liaoning', 'Jilin') THEN 'North east' ELSE [province] END`
- Calculate a measure by using a complex expression: order price per customer
`sum([order_amt])/count(distinct [customer_name])`
- Add a UNIX timestamp
`from_unixtime([order_id] + 1234567890)`
- Locate different days in a month
`day([order date])`
Returns a number in the range of 1 to 31.
- Locate different hours in a day
`hour([report_date])`
Returns a number in the range of 0 to 23.
- Calculate the advertisement conversion rate
`CASE WHEN sum([Views]) > 0 THEN sum([Conversion times])/sum([Views]) ELSE 0 END`
The following example is an incorrect expression: `sum(CASE WHEN [Views] > 0 THEN [Conversion times]/[Views] ELSE 0 END)`. For metrics that indicate rates, you must perform the sum operation before the division operation.

7.4.3.4.6.5. Add a calculated field

This topic describes how to add a calculated field.

Prerequisites

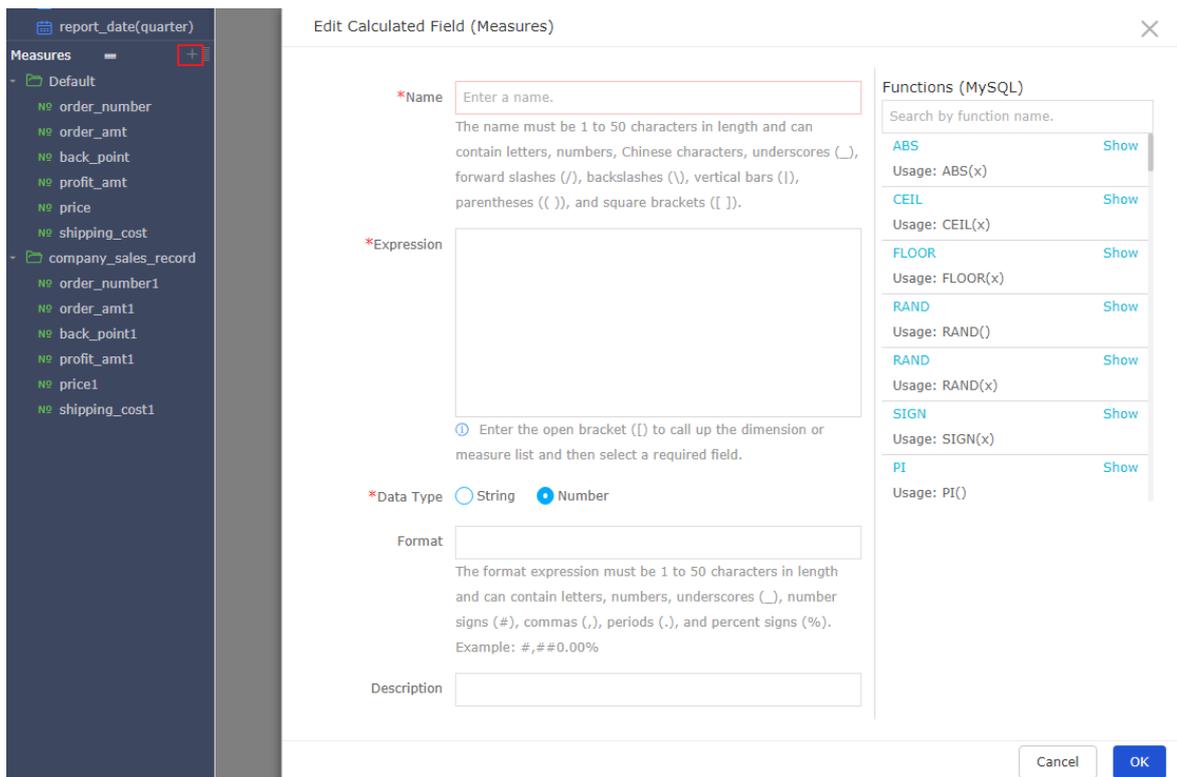
For more information about the usage and expressions of calculated fields, see [Rules for using calculated fields](#) and [Expressions of calculated fields](#).

Context

The following example uses the company_sales_record dataset to calculate the average profit of orders.

Procedure

1. Log on to the Quick BI console.
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the Datasets page, click the company_sales_record dataset.
5. In the Measures list, click the  icon. The **Edit Calculated Field (Mea.)** dialog box appears.



6. Enter the measure name and expression.

If you want to calculate the average profit of orders, enter an expression to divide the total profit of the orders by the order quantity.

Edit Calculated Field (Measures) ✕

***Name**
The name must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (`_`), forward slashes (`/`), backslashes (`\`), vertical bars (`|`), parentheses (`()`), and square brackets (`[]`).

***Expression**
Enter the open bracket (`[`) to call up the dimension or measure list and then select a required field.

***Data Type** String Number

Format
The format expression must be 1 to 50 characters in length and can contain letters, numbers, underscores (`_`), number signs (`#`), commas (`,`), periods (`.`), and percent signs (`%`).
Example: `#,##0.00%`

Description

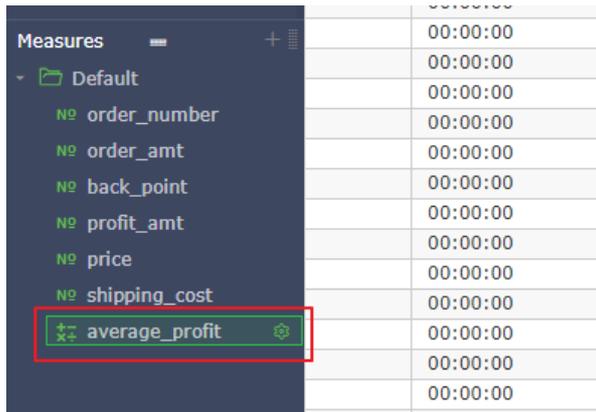
Functions (MySQL)

Search by function name.

ABS	Show
Usage: ABS(x)	
CEIL	Show
Usage: CEIL(x)	
FLOOR	Show
Usage: FLOOR(x)	
RAND	Show
Usage: RAND()	
RAND	Show
Usage: RAND(x)	
SIGN	Show
Usage: SIGN(x)	
PI	Show
Usage: PI()	

? **Note** You must use an English IME to enter the expression.

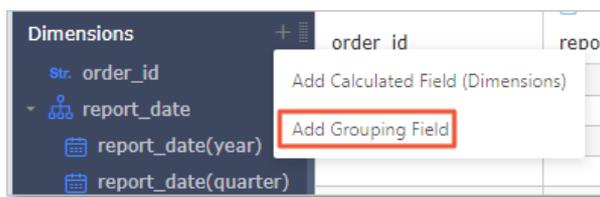
7. Select a data type. For example, if the average profit is a numeric value, you need to select **Number** for Data Type.
8. Click **OK** to add the field.
9. Click **Save** in the upper-right corner to save the dataset.
10. Click **Refresh Preview** to view the new calculated field.



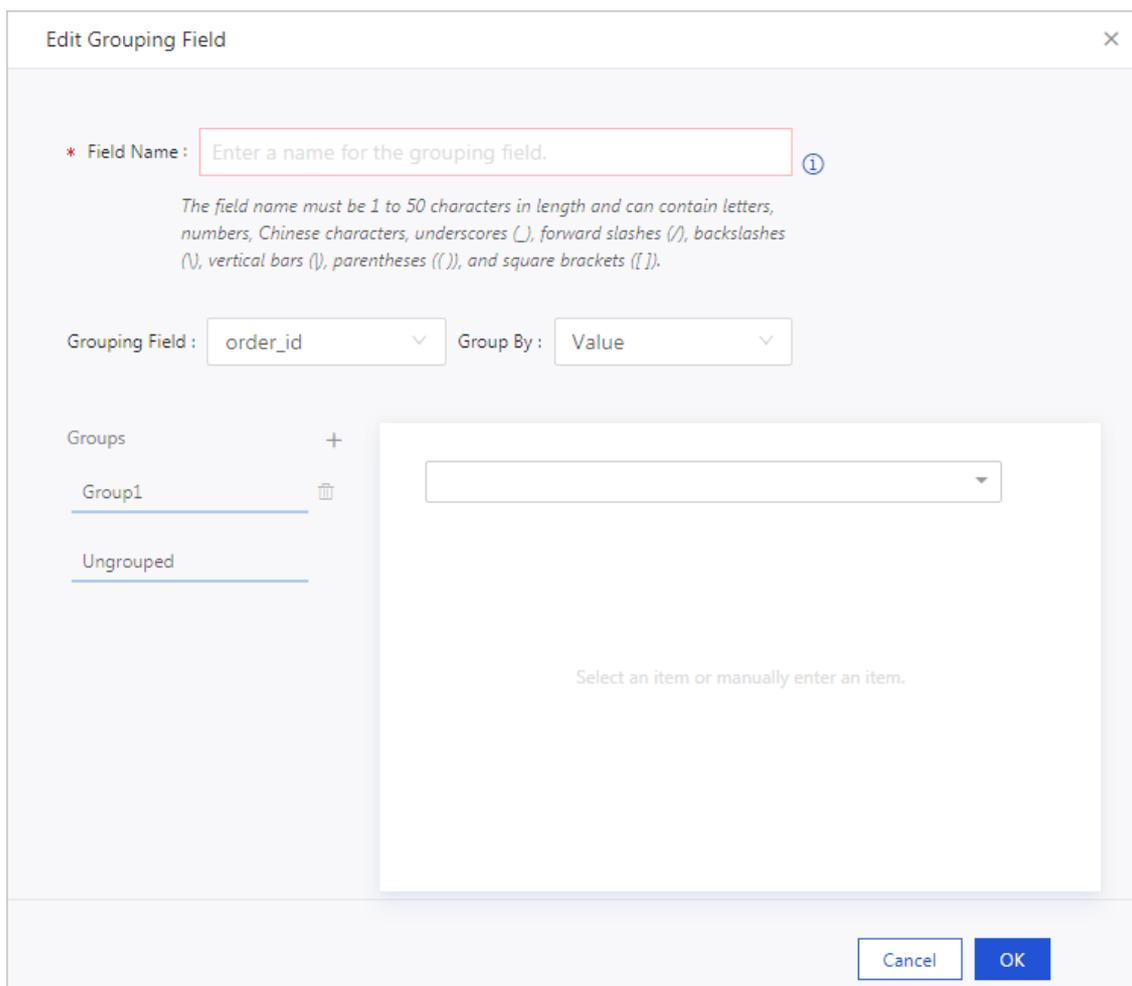
7.4.3.4.7. Add a grouping field

On the dataset edit page, you can use the **Add Grouping Field** feature to group data and store group information.

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the Datasets page, click the `company_sales_record` dataset.
5. On the dataset edit page, click the  icon next to **Dimensions** and select **Add Grouping Field**.



6. In the **Edit Grouping Field** dialog box, enter the required information and click **OK**.



7. Click **Save** and then click **Refresh Preview**. The dimension list shows the new grouping field.

7.4.3.5. Rename a dataset

This topic describes how to rename a dataset.

Prerequisites

A dataset is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the Datasets page, find the required dataset, click the  icon in the **Actions** column, and then select **Edit Properties**.
5. On the **Edit Properties** page, modify the dataset name and click **Save**.



7.4.3.6. Search for a dataset

This topic describes how to search for a dataset.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the **Datasets** page, find the search box.
5. Enter a keyword of the target dataset in the search box and click the **Q** icon.



7.4.3.7. Transfer a dataset

This topic describes how to transfer a dataset to another user.

Prerequisites

The Quick BI service is purchased.

Context

You can transfer a dataset to another Apsara Stack tenant account or RAM user in the same workspace.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the **Datasets** page, find the required dataset, click the **⋮** icon in the **Actions** column, and then select **Edit Properties**. You can also right-click the required dataset and select **Edit Properties**.
5. In the **Edit Properties** panel, select a new owner and click **Save**.

Edit Properties

* Name: company_sales_record_en_us

Owner: [dropdown]

description: Please add an object description

Security Level: Private (Allow Only Workspace Owner to Edit)
 Protected (Allow Other Workspace Members to Edit)

7.4.3.8. Copy a dataset from one workspace to another

This topic describes how to copy a dataset from one workspace to another.

Prerequisites

The Quick BI service is purchased.

Context

Only group workspaces support this feature. Only an administrator of two workspaces can copy a dataset from one workspace to the other.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the **Datasets** page, right-click the dataset that you want to copy and select **Copy**.
5. In the **Copy Dataset** dialog box, configure the destination workspace, storage path, and the name of the dataset in the destination workspace.

Copy Dataset

Workspaces: [blurred]

Save To: MaxComputer

Datasets: odps_sql223

A name must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

Cancel OK

Note If the destination workspace does not contain the data source based on which the dataset is created, the data source is also copied to the destination workspace.

6. Click OK.

7.4.3.9. Configure the dataset cache

This topic describes how to configure and clear the cache of a dataset. After you enable dataset caching, when an SQL statement is executed within the cache duration, the system queries data from the cache.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

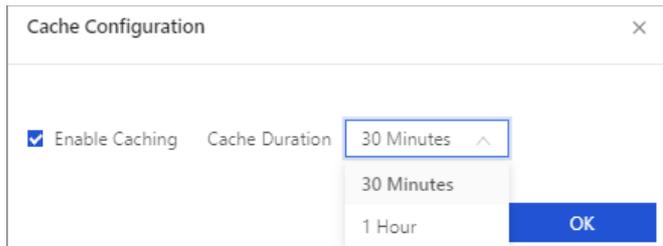
Context

After you enable caching, you can set the cache duration to 30 minutes or 1 hour.

- After the cache duration elapses, the cached data becomes invalid, and queries on the report page generate new cache. If the same SQL statement is executed again within the cache duration, the system queries data from the new cache.
- The cache is cleared based on datasets. When you trigger cache clearing for a dataset or after the cache duration elapses, the cache of all charts that are generated based on the dataset is cleared.

Configure the cache settings

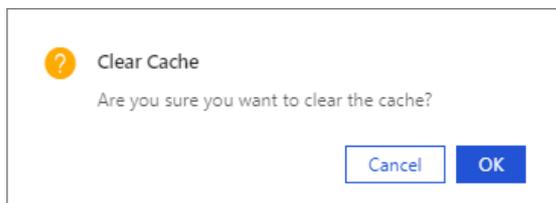
1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the Datasets page, find the required dataset, click the  icon in the **Actions** column, and then select **Cache configuration**.
5. In the **Cache configuration** dialog box, select **Cache configuration** and configure the cache time.



6. Click OK.

Clear the cache of a dataset

1. On the Datasets page, find the required dataset, click the  icon in the **Actions** column, and select **clear cache**.
2. In the clear cache message, click OK.



7.4.3.10. Create a dataset folder

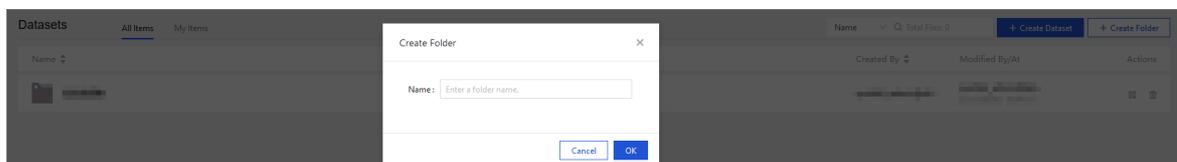
This topic describes how to create a dataset folder on the **Datasets** page.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. In the upper-right corner of the **Datasets** page, click **Create Folder**.
5. In the **Create Folder** dialog box, enter a name for the folder and click **OK**.



7.4.3.11. Rename a dataset folder

This topic describes how to rename a dataset folder.

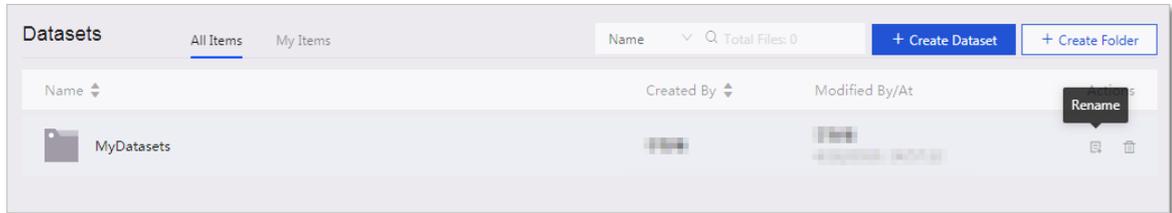
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)

2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the **All Items** tab of the **Datasets** page, find the folder that you want to rename.
5. Right-click the folder and select **Rename**, or click the  **Rename** icon in the Actions column.



6. In the Rename dialog box, enter a new name and click **OK**.

7.4.3.12. Delete a dataset

This topic describes how to delete a dataset and the common issues that may occur when you delete a dataset.

Prerequisites

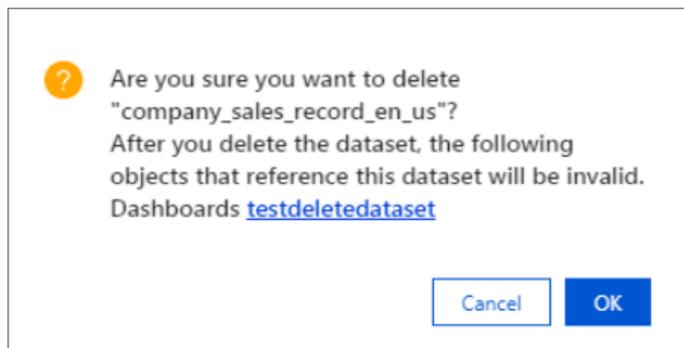
A dataset is created.

Context

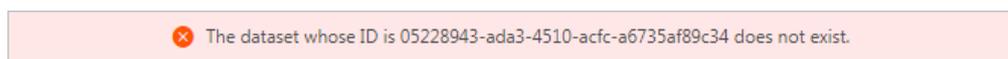
If workbooks are created based on the dataset, a notification is displayed when you attempt to delete the dataset, as shown in [Notification](#).

After you delete a dataset, an error message occurs when you access the dashboard created based on that dataset, as shown in [Error message](#).

Notification



Error message



Procedure

1. [Log on to the Quick BI console](#).
2. Click the **Workspace** tab.
3. In the left-side navigation pane, click **Datasets**.
4. On the **My Items** tab of the **Datasets** page, find the dataset that you want to delete.
5. Click the  icon in the **Actions** column and select **Delete**. You can also right-click the dataset and select

Delete.

7.4.3.13. Configure row-level permissions

Row-level permissions are granted based on datasets. Quick BI supports two authorization modes: user/user group-based authorization and tag-based authorization.

User/user group-based authorization is suitable for scenarios that involve a small number of members. Tag-based authorization is suitable for scenarios that involve a large number of members. Tag-based authorization allows you to authorize all users at the same time. In scenarios that involve a large number of members, this mode reduces the costs and complexity of row-level permission configurations and facilitates permission management. For information about how to configure row-level permissions, see [Manage row-level permissions](#).

7.5. Dashboards

7.5.1. Dashboard overview

7.5.1.1. Dashboard features

This topic describes the features of a Quick BI dashboard.

- Supports the Filter Bar, Text Area, IFrame, Tab, and Image widgets. You can drag and drop widgets to create pages for various products.
- Provides a wide range of chart components. You can easily create various reports by setting chart elements. A chart can be displayed in either standard or full-screen mode.
- Uses a more flexible tile layout. A dashboard visualizes data. It allows you to filter and query data, use multiple data display modes, and highlight the key fields of data.
- Enables you to drag, drop, and click fields in a dashboard to display data. You can follow the instructions on the pages to analyze data for better user experience.

7.5.1.2. Chart types and scenarios

Different types of data need to be displayed in charts of different types. Currently, Quick BI supports various data charts, such as line charts, vertical bar charts, bubble maps, and funnel charts.

For information about how to create charts, see [Create a dashboard](#).

The following table describes the analysis types and scenarios for each chart type.

Chart types and scenarios

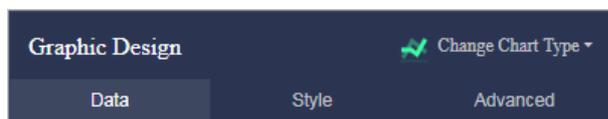
Analysis type	Description	Scenario	Applicable chart
Comparison	Compares values or compares measures by category	Compares the sales or income between different countries or regions.	Vertical bar chart, radar chart, funnel chart, cross table, polar diagram, butterfly chart, and word cloud
Percentage	Displays the proportion of a part to the total, or the proportion of a certain value to the whole.	Displays the sales of the salesperson that has the greatest percentage of total sales.	Pie chart, funnel chart, gauge, and treemap

Analysis type	Description	Scenario	Applicable chart
Relevance	Displays the relationship between values or measures.	Displays the relationship between two measures. This helps you understand the influence the first measure has on the second measure.	Scatter chart, treemap, kanban, hierarchy chart, and flow analysis chart
Trend	Displays data trends, especially trends based on time, such as year, month, or day, or the progress of a data metric or other possible patterns.	Displays trends in sales or revenue for a product over a period of time.	Line chart
Geographic map	Displays the size and distribution of data metrics of a country or region on a map. The datasets must contain geographic data.	Displays the income information about different regions of a country.	Bubble map and colored map

7.5.1.3. Data elements of a chart

This topic describes the data elements of a chart.

The Graphic Design section for each chart has three tabs: **Data**, **Style**, and **Advanced**.



- On the **Data** tab, you can specify the data you want to display in the chart.
- On the **Style** tab, you can set the chart layout and details of the chart.
- On the **Advanced** tab, you can configure the filter interaction feature to dynamically compare and display data.

Each chart has unique core data elements. For example, in a geo map, a latitude field is required. Otherwise, data cannot be displayed.

The following table lists the core data elements of each chart type.

Data elements of different types of charts

Chart type	Required data element	Data element description
Line chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.

Chart type	Required data element	Data element description
Stacked line chart	Category axis and value axis	You must specify at least one dimension for the category axis and at least one measure for the value axis.
Area chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Stacked area chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one dimension for the legend.
100% stacked area chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Vertical bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Stacked vertical bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
100% stacked vertical bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Circular bar	Category axis and value axis	You can specify only one dimension for the category axis and only one measure for the value axis.
Horizontal bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Stacked horizontal bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.

Chart type	Required data element	Data element description
100% stacked horizontal bar chart	Category axis, value axis, and legend	You must specify at least one dimension for the category axis and at least one measure for the value axis. You can specify only one measure for the legend.
Waterfall chart	Category axis and value axis	You must specify at least one dimension for the category axis and at least one measure for the value axis.
Combination chart	Category axis and value axis	You must specify at least one dimension for the category axis and at least one measure for the value axis.
Pie chart	Labels and central angles	You can specify only one dimension for labels and only one measure for the central angles.
Cross table	Rows and columns	You can specify an unlimited number of dimensions for the rows and an unlimited number of measures for the columns.
Pivot table	Rows and values	You can specify an unlimited number of dimensions for the rows and an unlimited number of measures for the values.
Bubble map	Geo locations, bubble size, and bubble color	You can specify only one dimension for geo locations. The data type of the dimension must be Geo. You can specify one to five measures for the bubble size and only one measure for the bubble color.
Colored map	Geo locations and colorscale	You can specify only one dimension for geo locations. The dimension type must be Geo. You can specify one to five measures for the colorscale.
Gauge	Pointer angle	You can specify only one measure for the pointer angle.
Liquid fill chart	Pointer	You must specify one to five measures for the pointer.
Progress bar	Pointer	You must specify one to five measures for the pointer.
Radar chart	Labels and length	You can specify one to two dimensions for labels and at least one measure for the length.

Chart type	Required data element	Data element description
Scatter chart	Color, category, x-axis, and y-axis	You can specify only one dimension or measure for color. You can specify only one measure for the x-axis, only one measure for the y-axis, and only one dimension for the category.
Bubble chart	x-axis, y-axis, color, category, and size	You can specify only one dimension or measure for the x-axis, only one measure for the y-axis, and only one measure or dimension for color. You can specify only one dimension for the category and only one measure for the size.
Facet scatter chart	x-axis, y-axis, and legend	You can specify one to three measures for the x-axis. You can specify only one measure for the y-axis and only one dimension for the legend.
Funnel chart	Funnel layering and tier area	You can specify one to ten measures for the tier area and only one dimension for the funnel layering
Kanban	Labels and metrics	You can specify only one dimension for labels and 1 to 10 measures for metrics.
Treemap	Labels and size	You can specify only one dimension for labels and only one measure for the size.
Polar diagram	Labels and arc radius	You can specify only one dimension for the label and only one measure for the arc radius.
Word cloud	Word size and words	You can specify only one measure for the word size and only one dimension for words.
Butterfly chart	Comparison metrics and tier area	You can specify one to ten measures for the tier area. You must specify at least one dimension for the comparison metrics.
Hierarchy chart	Node labels and node metrics	You must specify at least two dimensions for node labels and at least one measure for node metrics.
Flow Analysis	Central node, node type, node name, and node indicator	You can specify only one dimension or one measure for each data element.

Chart type	Required data element	Data element description
Waterfall chart	Category axis and value axis	You must specify at least one dimension for the category axis. You can specify only one measure for the value axis.
Indicator trend graph	Date and indicator	You can specify only one dimension for the date. You must specify at least one measure for the indicator.
Sankey diagram	Node type and node height	You can specify two to five dimensions for the node type and only one measure for the node height.
Ranking board	Category and indicator	You can specify only one dimension for the category and only one measure for the indicator.
Flop	Display metrics	You can specify only one measure for the display metrics.

7.5.2. Access a dashboard

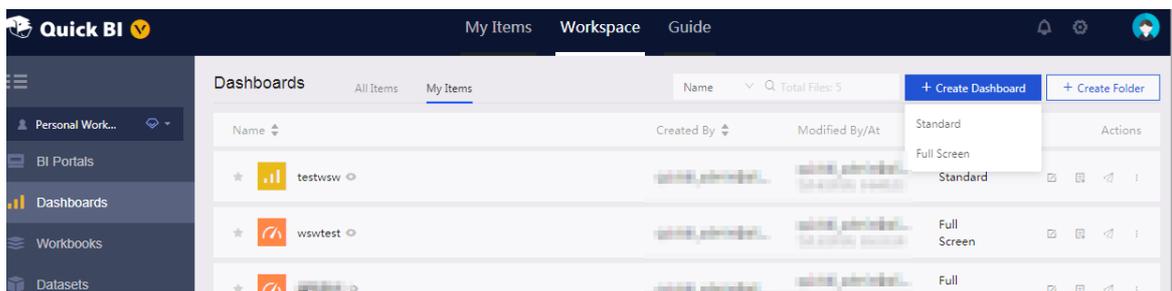
This topic describes how to access a dashboard.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. Click the **Workspace** tab.
3. In the left-side navigation pane of the Workspace page, click **Dashboards**.
4. Choose **Create Dashboard > Standard** to go to the dashboard.



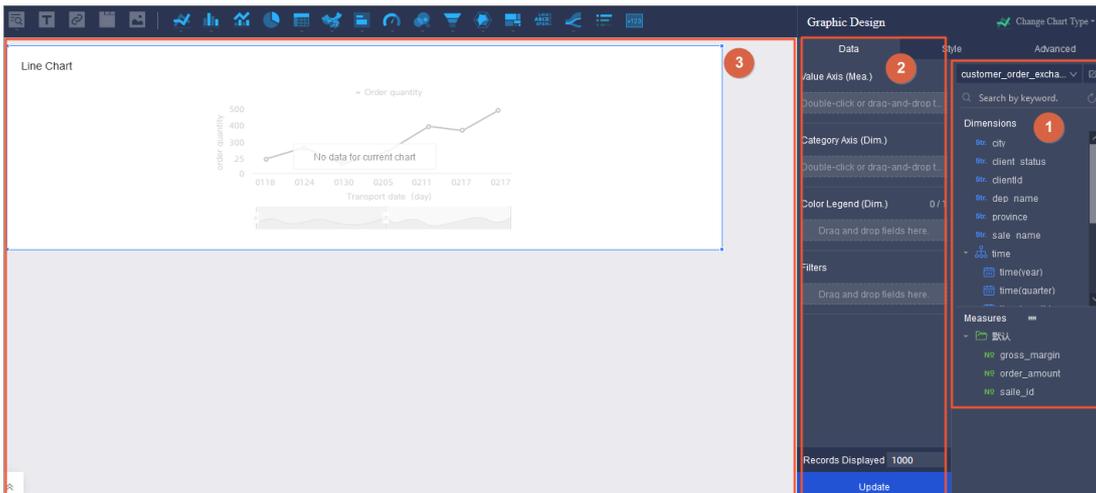
7.5.3. Areas of a dashboard

7.5.3.1. Overview

This topic describes the functional sections of the dashboard edit page.

The dashboard edit page consists of three sections, as shown in [Dashboard](#).

Dashboard



- The dataset selection section is marked as 1. In this section, you can switch the current dataset to another. The fields of a dataset are displayed in the Dimensions and Measures lists based on the data types preconfigured in the system. You can select dimensions and measures based on the chart type.
- The dashboard configuration section is marked as 2. In this section, you can select a chart type, and edit the title, layout, and legend position of a chart. On the Advanced tab, you can associate the current chart with other charts and display analysis results from multiple perspectives. You can also filter data by using filters or insert a query control to query key data in a chart.
- The dashboard display section is marked as 3. In this section, you can drag charts to adjust their positions and change chart types. For example, you can change a bar chart to a bubble map. The system displays information about missing or error elements. In the dashboard display section, you can save, preview, or create a dashboard. The dashboard provides instructions to help you learn how to create a dashboard.

7.5.3.2. Dataset selection area

7.5.3.2.1. Switch datasets

This topic describes how to switch datasets in a dashboard.

Prerequisites

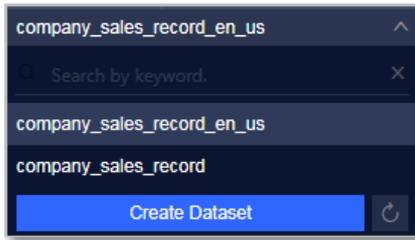
The Quick BI service is purchased.

Context

If you cannot find the required dataset in the drop-down list, go back to the dataset management page and make sure that you created the dataset. For information about how to create a dataset, see [Create a dataset](#).

Procedure

1. [Access a dashboard](#).
2. On the Data tab, click the **Switch Datasets** icon.
3. Select the required dataset from the drop-down list that appears.



7.5.3.2.2. Search for a dimension or measure

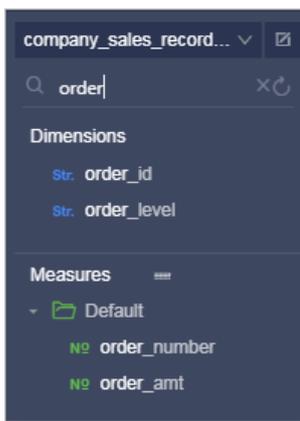
This topic describes how to search for a dimension or measure.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Access a dashboard](#).
2. On the **Data** tab in the **Graphic Design** section, find the search box and enter the keyword of the field, such as **order**.
3. Click the  icon.



For information about how to edit dimensions and measures, see [Edit a dimension](#) and [Edit a measure](#).

7.5.3.3. Dashboard graphic design area

7.5.3.3.1. Select fields

This topic describes how to select fields from a dataset.

Prerequisites

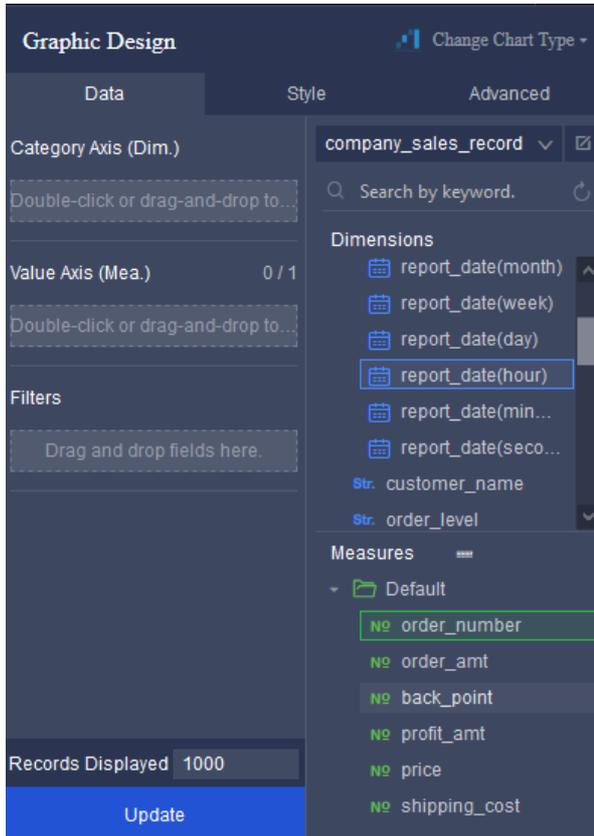
- The Quick BI service is purchased.
- A dataset is selected in the dataset selection area and is edited as needed. For information about how to edit a dataset, see [Edit a dataset](#).

Context

Procedure

1. Access a dashboard.
2. Select a chart from the toolbar on the top of the dashboard.
3. Click the chart icon. The chart appears in the display area of the dashboard. If you want to change the chart type, click **Change Chart Type** in the Graphic Design area, and select the required chart type.
4. On the **Data** tab, select the required fields, as shown in **Select fields**. Double-click a field to add it to the Dimensions or Measures list, or drag the field to the display section.

Select fields



- If you want to delete a field, click the  icon next to the field.
 - Click the  icon next to the dimension field. You can drill down to change the dimension hierarchy and the granularity of the analysis.
 - Click the  icon next to a dimension field to sort dimension fields.
 - You can click the  icon next to the field and select the method as required. The aggregation methods include SUM, COUNT, MAX, MIN, AVG, COUNT DISTINCT, median, percentile, variance, standard deviation, and compare operations.
5. Click **Update**. The chart is updated.

7.5.3.3.2. Use the color legend

This topic describes how to use the color legend.

Prerequisites

- The Quick BI service is purchased.

- A dataset is selected in the dataset selection area and is edited as needed. For information about how to edit a dataset, see [Edit a dataset](#).

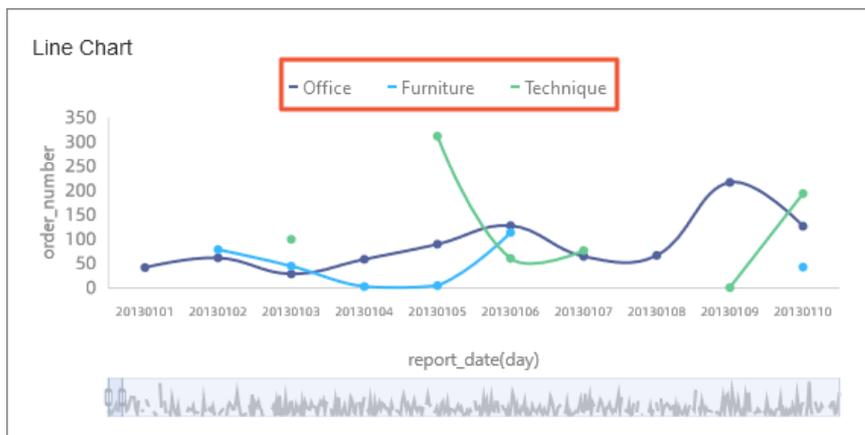
Context

The color legend feature displays the values of a selected field in different colors in a chart.

Only dimensions can be added to the Color Legend field.

Procedure

1. [Access a dashboard](#).
2. Drag a dimension, such as product_type, to the **Color Legend (Dim.)** field.
3. Click **Update**. The values of the field are displayed in different colors in the chart.



4. Change the legend color of product_type in the **Style > Series Settings** section.

7.5.3.3.3. Sort field data

This topic describes how to sort data on the **Data** tab.

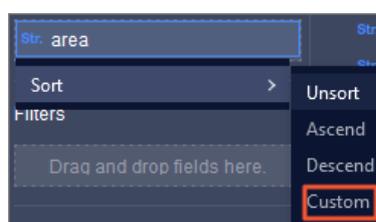
Prerequisites

- The Quick BI service is purchased.
- A dataset is selected in the dataset selection area and is edited as required. For information about how to edit a dataset, see [Edit a dataset](#).

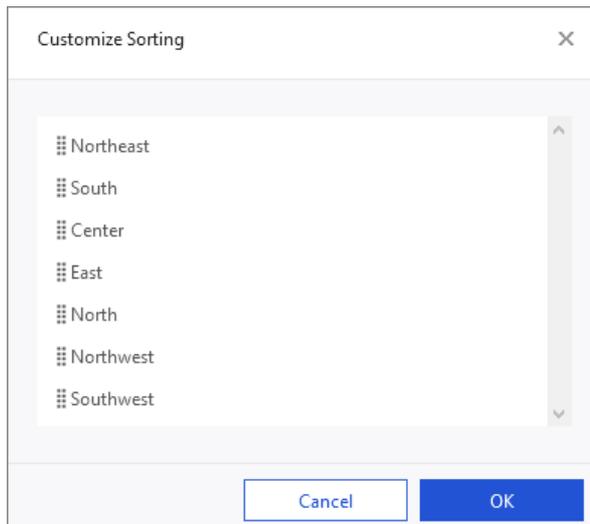
Procedure

1. [Go to the dashboard edit page](#).
2. Select a field and sort data based on the field.
 - If a dimension field is selected, perform the following steps:
 - a. Click the upward arrow next to the field, select **Sort**, and then select **Unsort**, **Ascend**, **Descend**, or **Custom**. In this example, **Custom** is selected for the region field, as shown in [Custom sorting](#).

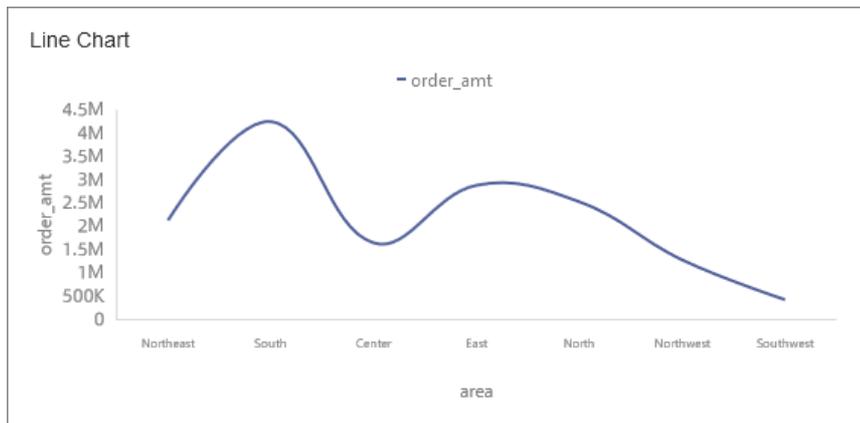
Custom sorting



- b. On the custom sorting page, move the pointer over a region, drag the region to adjust its sequence, and then click **OK**.



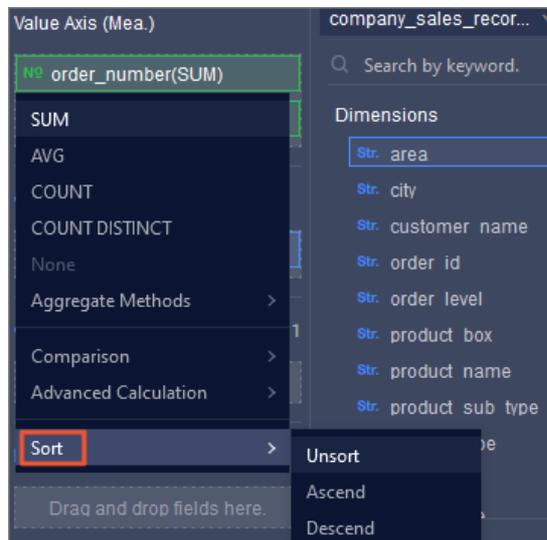
- c. After the configuration is complete, click **Update**.



- o If a measure field is selected, perform the following steps:

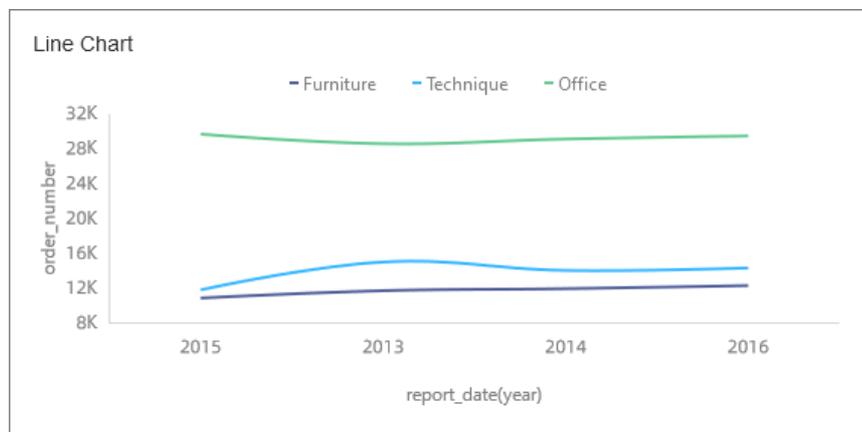
- a. Click the upward arrow next to the field, select **Sort** and then select Unsort, Ascend, Descend, or Custom. In this example, Ascend is selected for the order_number field, as shown in [Specify the sorting order](#).

Specify the sorting order



Note You can also sort data based on the comparison results and advanced calculation results.

- b. Click **Update**.



7.5.3.3.4. Configure advanced calculations

This topic describes how to perform advanced calculations on data of a specified field on the **Data** tab.

Prerequisites

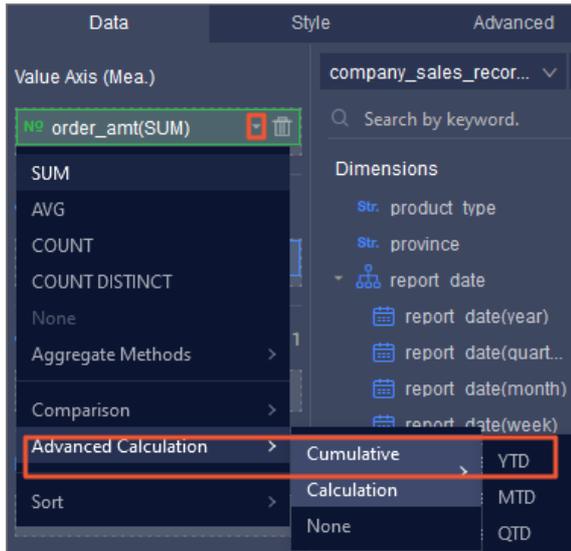
The Quick BI service is purchased.

Procedure

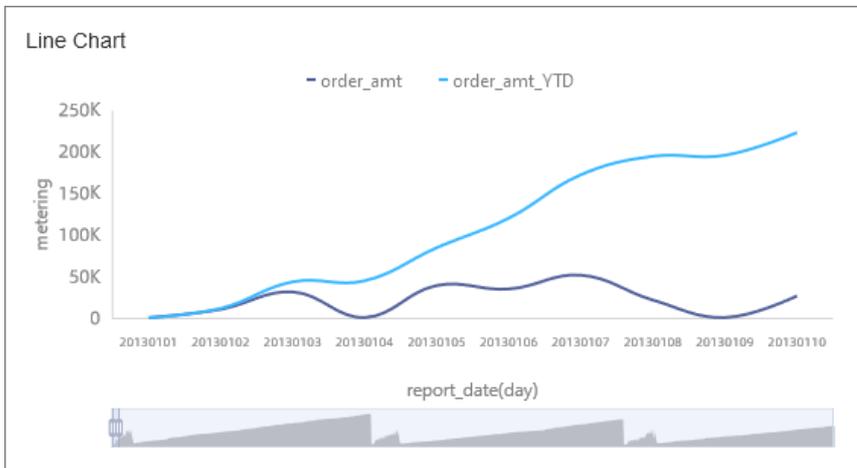
1. [Access the required dashboard](#).
2. In the dataset selection section, select a dataset and add the related fields to the measure and dimension fields. Make sure that the dataset has been edited as required. For more information, see [Edit a dataset](#).

3. Select a field, for example, order_amt.
4. Click the arrow icon next to the field, choose **Advanced Calculation > Cumulative**, and then select YTD, MTD, or QTD as required, as shown in [Configure advanced calculations](#).
 - o YTD: The value is accumulative for each year. The value is accumulated from the first day to the last day of the current year, and accumulation restarts at the beginning of the next year. In this example, the order price is accumulated from January 1, 2020 to the current time.
 - o MTD: The value is accumulative for each month.
 - o QTD: The value is accumulative for each quarter.

Configure advanced calculations



5. Click **Update**.



7.5.3.3.5. Configure the comparison feature

This topic describes how to configure the comparison feature for data of a specified field on the Data tab.

Prerequisites

- The Quick BI service is purchased.
- A dataset is selected in the dataset selection area and is edited as needed. For information about how to edit a dataset, see [Edit a dataset](#).

- A dashboard is created.
- A query control is added. For more information, see [Create a filter bar](#).

Context

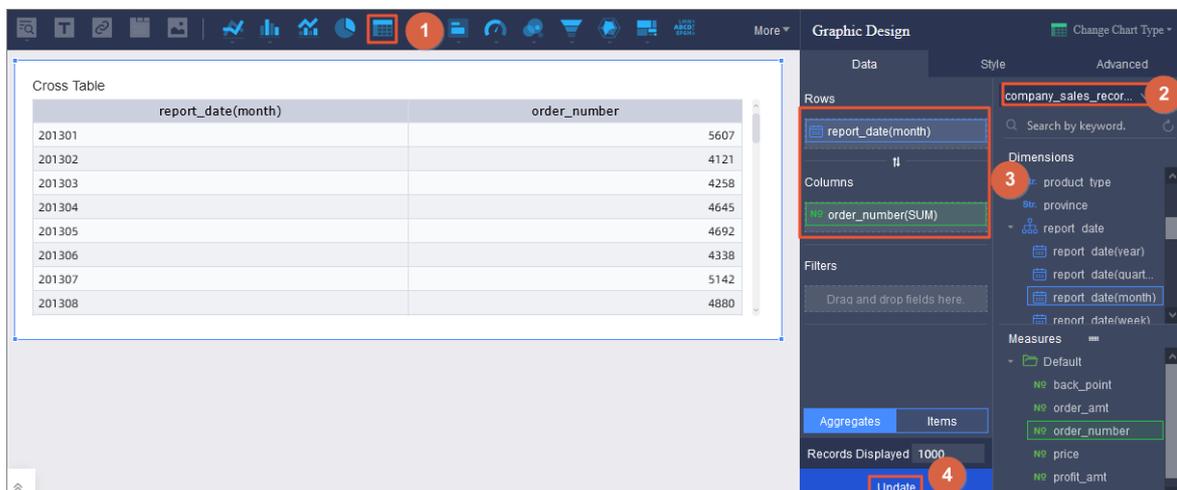
- If **Advanced Calculation** is configured for a measure, **Comparison** is unavailable.
- If **Comparison** is configured for a measure, **Advanced Calculation** is unavailable.

One DATE field in dimensions

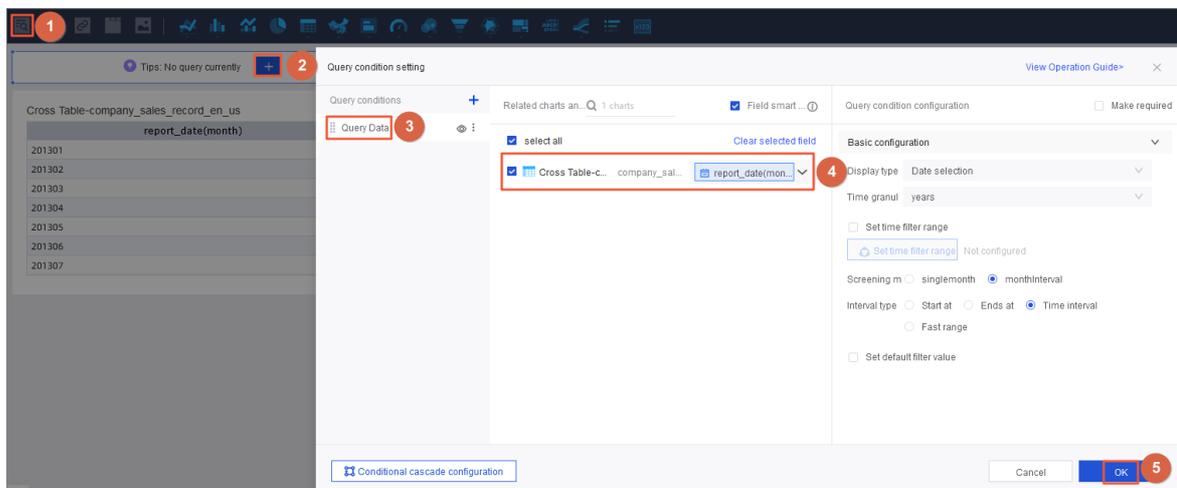
If dimensions contain only one DATE field, set the granularity for the comparison feature to the granularity of this field.

The following example demonstrates how to configure the comparison feature for a measure. A cross table is used in this example.

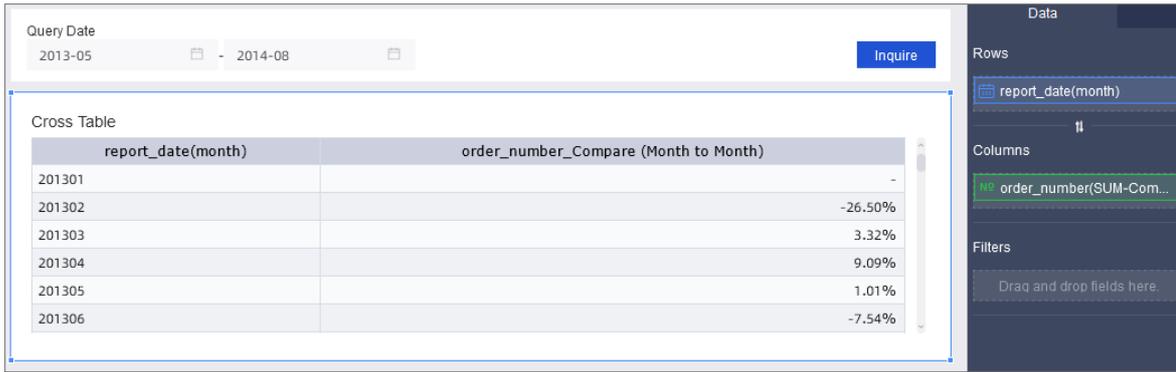
1. [Go to the dashboard edit page](#). This topic describes how to create a standard dashboard.
2. On the dashboard edit page, add a cross table and configure it, as shown in the following figure. In this example, the DATE field `order_date(month)` is added to the Rows field.



3. Move the pointer over the required measure in the Columns field and click the icon next to the measure.
4. Choose **Comparison > Compare (Month to Month)**.
5. Add a query control and configure it, as shown in the following figure.



6. In the query control, specify a time range and click **Inquire**.

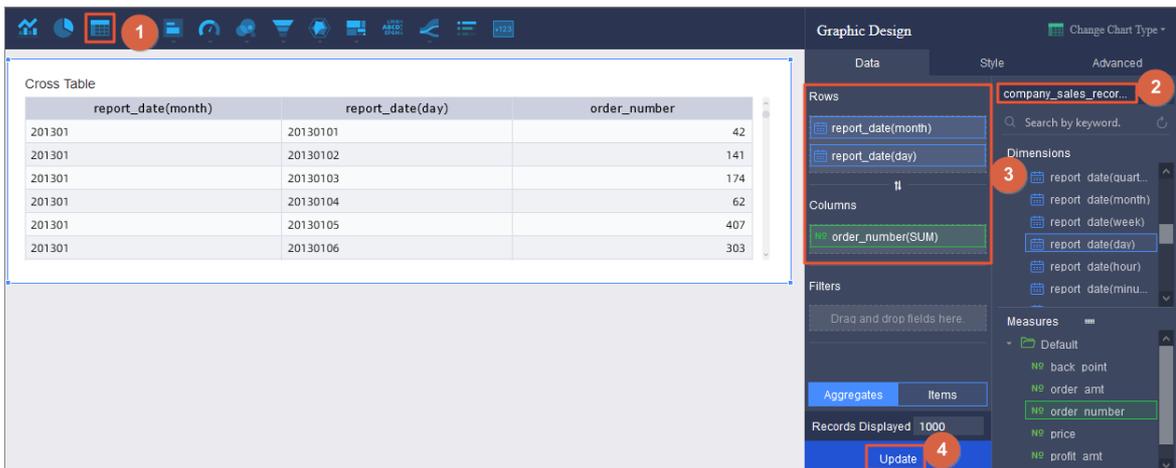


Multiple DATE fields in dimensions

If dimensions contain multiple DATE fields, the field with the smallest granularity is used to configure the comparison feature.

The following example demonstrates how to configure the comparison feature for a measure. A cross table is used in this example.

1. On the dashboard edit page, add a cross table and configure it, as shown in the following figure. In this example, DATE fields order_date(day) and order_date(month) are added to the Rows column.

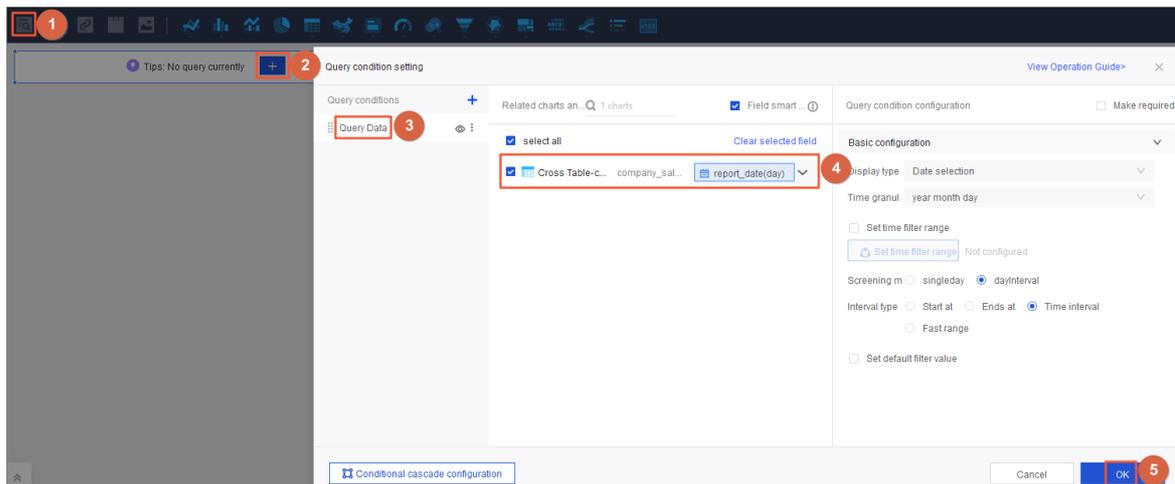


2. Move the pointer over the required measure in the Columns field and click the  icon next to the measure.
3. Choose **Comparison > Custom**.
4. In the **Comparison Settings** dialog box, configure the parameters listed in the following table.

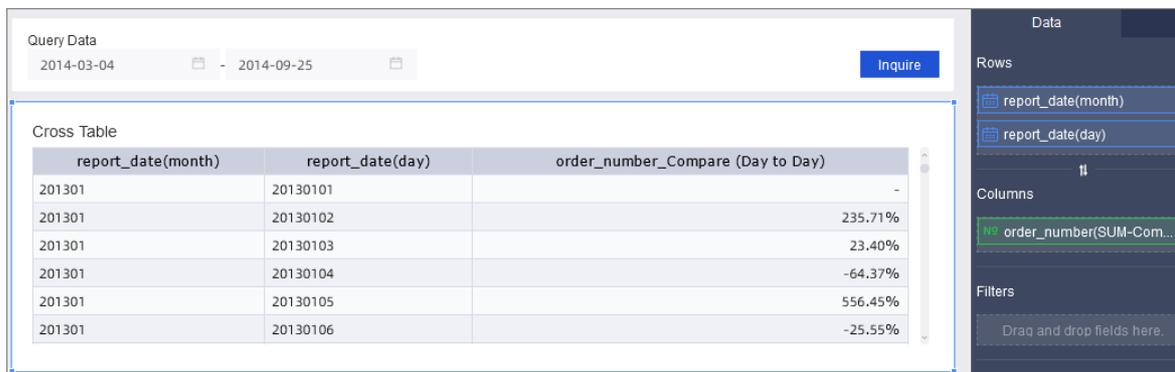
Parameter	Value	Description
Comparison Date	order_date(day)	If dimensions contain multiple DATE fields, the field with the smallest granularity is used and cannot be modified. In this example, the value is order_date(day) .
Comparison Type	<ul style="list-style-type: none"> ◦ Compare (Day to Day) ◦ Compare (of Last Week) ◦ Compare (of Last Month) ◦ Compare (of Last Year) 	The valid values of Comparison Type vary based on the setting of Comparison Date . In this example, set the value to Compare (Day to Day) .

Parameter	Value	Description
Data Settings	<ul style="list-style-type: none"> Specific Value Difference Percentage Difference 	If you set the value to Percentage Difference , you must click Advanced Settings and select a formula.
Formula	<ul style="list-style-type: none"> (Value of Current Period/Value of Previous Period - 1) × 100% (default value) 1 - (Value of Current Period/Value of Previous Period) × 100% Displayed As - 	The formula takes effect only if Data Settings is set to Percentage Difference and the value of the previous period is less than 0.

- Click **OK**.
- Add a query control and configure it, as shown in the following figure.



- In the query control, specify a time range and click **Inquire**.

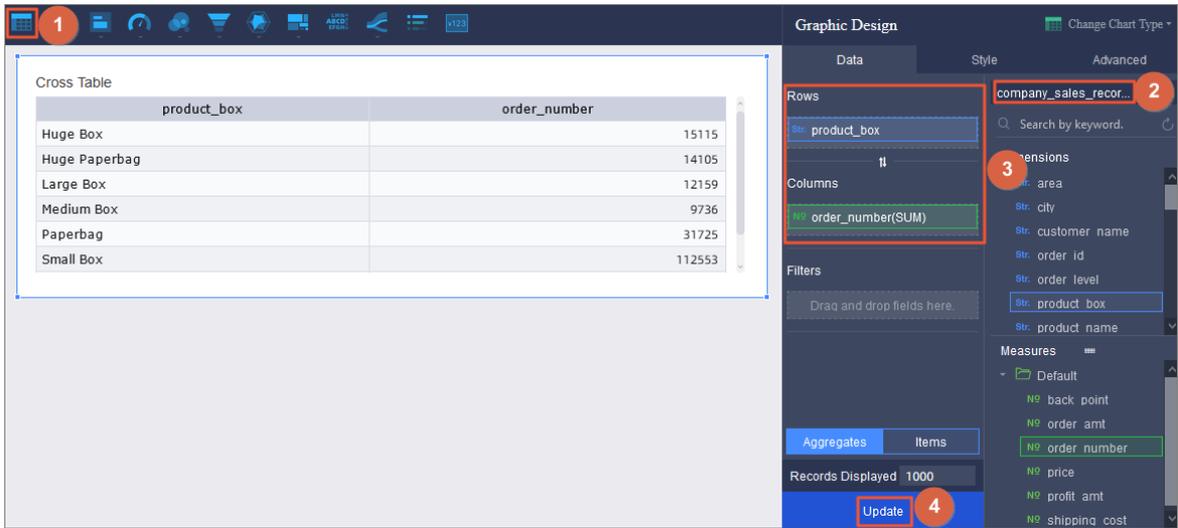


No DATE field in dimensions (year-to-year comparison)

If dimensions do not contain DATE fields and you want to configure the comparison feature for the chart, select a DATE field in a query control that is associated with the chart. Then, set Comparison Date to a field with the same granularity as this DATE field in the **Comparison Settings** dialog box.

The following example demonstrates how to configure year-to-year comparison for a measure. A cross table is used in this example.

1. On the dashboard edit page, add a cross table and configure it, as shown in the following figure.

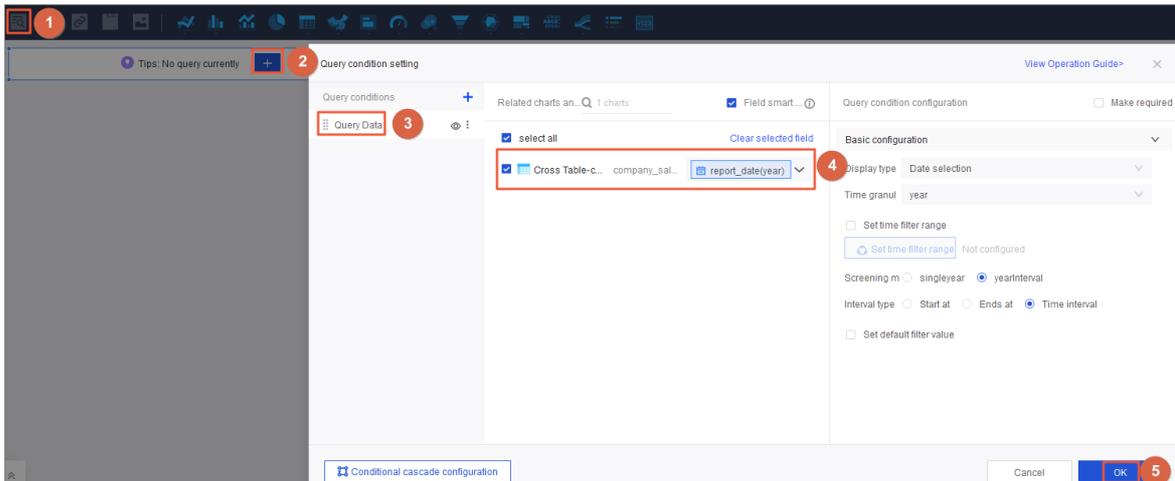


2. Move the pointer over the required measure in the Columns field and click the  icon next to the measure.
3. Choose **Comparison > Custom**.
4. In the **Comparison Settings** dialog box, configure the parameters listed in the following table.

Parameter	Value	Description
Comparison Date	order_date(year)	Configure this parameter as required. In this example, set the value to order_date(year) .
Comparison Type	<ul style="list-style-type: none"> ◦ Compare(Year to Year) ◦ Dynamic Ratio 	The valid values of Comparison Type vary based on the setting of Comparison Date . In this example, set the value to Compare (Year to Year) .
Data Settings	<ul style="list-style-type: none"> ◦ Specific Value ◦ Difference ◦ Percentage Difference 	If you set the value to Percentage Difference , you must click Advanced Settings and select a formula. In this example, set the value to Percentage Difference .
Formula	<ul style="list-style-type: none"> ◦ (Value of Current Period/Value of Previous Period - 1) × 100%: (default value) ◦ 1 - (Value of Current Period/Value of Previous Period) × 100% ◦ Displayed As - 	The formula takes effect only if Data Settings is set to Percentage Difference and the value of the previous period is less than 0.

5. Click **OK**.

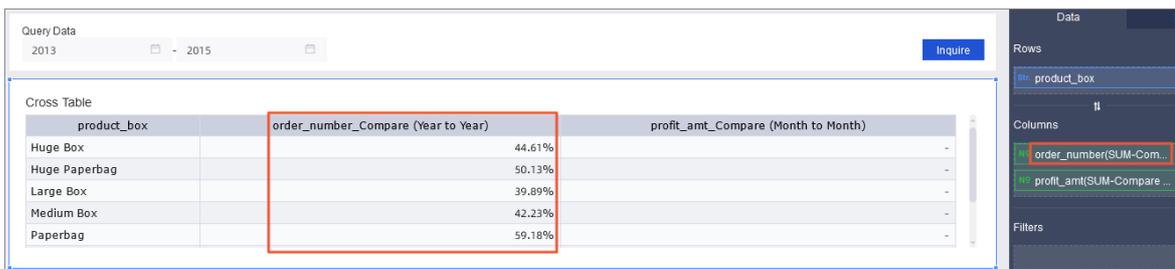
6. Add a query control and configure it, as shown in the following figure.



7. In the query control, specify a time range and click Inquire.



If the query conditions in a query control contain multiple DATE fields and the comparison feature is configured for multiple measures, the comparison feature takes effect only for the first measure.



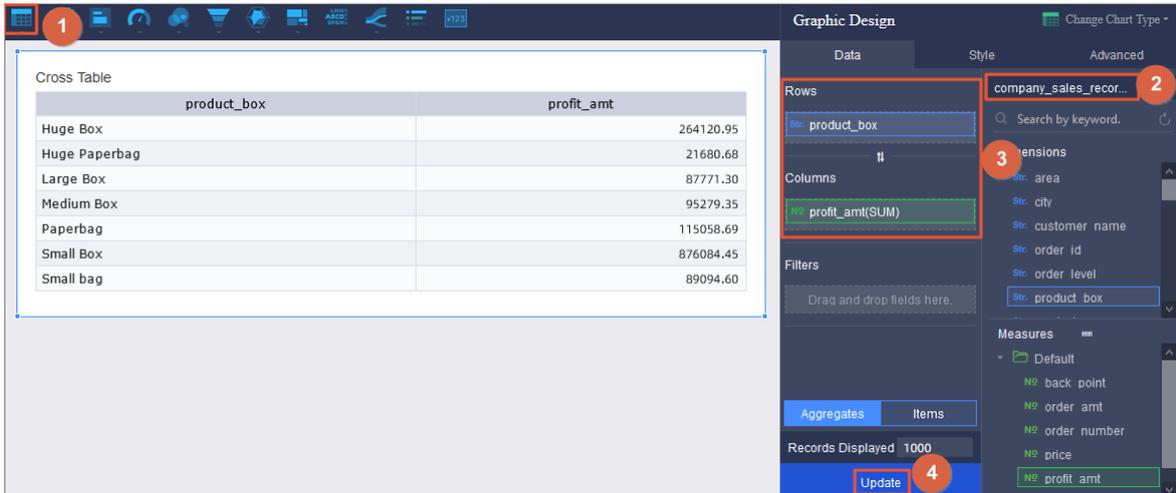
Note If the comparison result cannot be calculated, a hyphen (-) is displayed.

No DATE field in dimensions (Dynamic Ratio)

If dimensions do not contain DATE fields, you must select a DATE field in the **Comparison Settings** dialog box to configure the comparison feature.

You can also select **Dynamic Ratio**. The following example demonstrates how to configure the dynamic ratio. A cross table is used in this example.

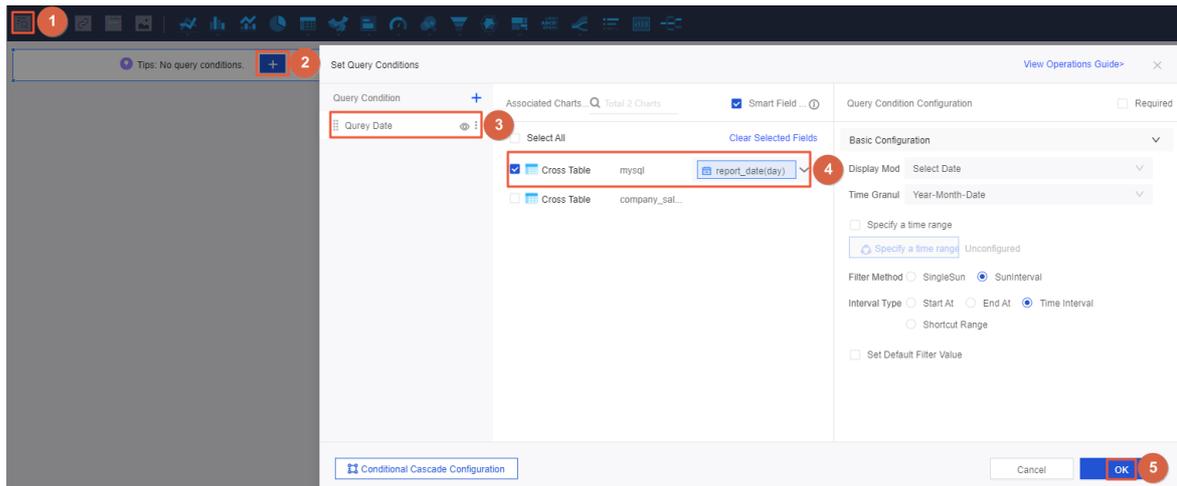
1. On the dashboard edit page, add a cross table and configure it, as shown in the following figure.



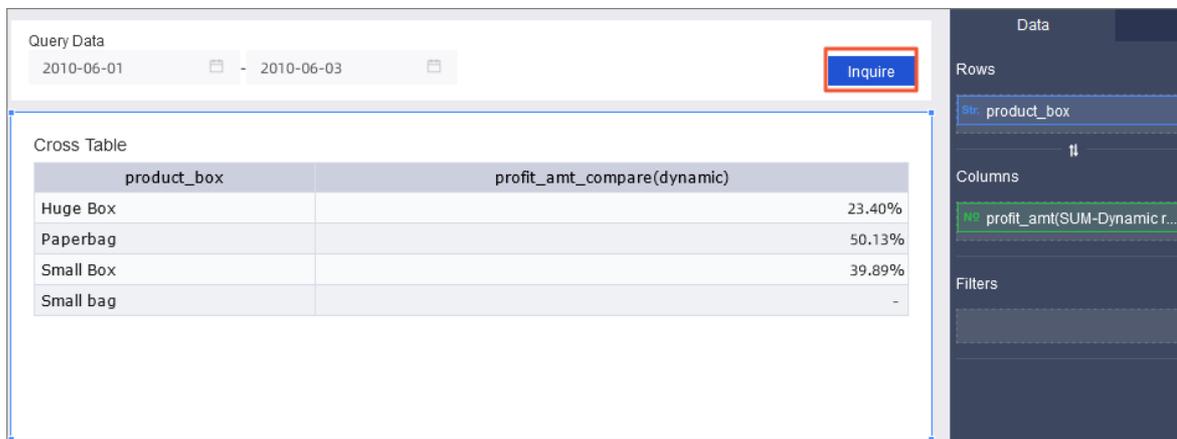
2. Move the pointer over the required measure in the Columns field and click the  icon next to the measure.
3. Choose **Comparison > Custom**.
4. In the **Comparison Settings** dialog box, configure the parameters listed in the following table.

Parameter	Value	Description
Comparison Date	order_date(day)	Configure the parameter as required. In this example, set the value to order_date(day) .
Comparison Type	<ul style="list-style-type: none"> ◦ Compare (Day to Day) ◦ Compare (of Last Week) ◦ Compare (of Last Month) ◦ Compare (of Last Year) ◦ Dynamic Ratio 	The valid values of Comparison Type vary based on the setting of Comparison Date . In this example, set the value to Dynamic Ratio .
Data Settings	<ul style="list-style-type: none"> ◦ Specific Value ◦ Difference ◦ Percentage Difference 	If you set the value to Percentage Difference , you must click Advanced Settings and select a formula. In this example, set the value to Percentage Difference .
Formula	<ul style="list-style-type: none"> ◦ (Value of Current Period/Value of Previous Period - 1) × 100% (default value) ◦ 1 - (Value of Current Period/Value of Previous Period) × 100% ◦ Displayed As - 	The formula takes effect only if Data Settings is set to Percentage Difference and the value of the previous period is less than 0.

5. Click **OK**.
6. Add a query control and configure it, as shown in the following figure.



7. In the query control, specify a time range and click **Inquire**.



8. View SQL statements. If **Comparison Type** is set to **Dynamic Ratio**, the comparison results are dynamically calculated based on the specified time range. The system calculates the difference before the specified date, and then subtracts the date that corresponds to the minus loop. For example:
- If the specified date is January 10, 2020, the date for comparison is January 9, 2020.
 - If the specified time range is January 1, 2020 to January 10, 2020, the time range for comparison is December 22, 2019 to December 31, 2019.
 - i. Click the  icon in the upper-right corner of the cross table.

ii. Select **View SQL Statements**. You can view the time ranges for comparison.

```
SQL
SELECT
  COMPANY_T_1_."product_box" AS A_PRODUCT_2_,
  SUM(COMPANY_T_1_."profit_amt") AS A_PROFIT_3_
FROM
  "public"."company_sales_record" AS COMPANY_T_1_
WHERE
  COMPANY_T_1_."report_date" >= TO_TIMESTAMP('2010-06-01 00:00:00', 'yyyy-MM-dd hh24:mi:ss')
  AND COMPANY_T_1_."report_date" <= TO_TIMESTAMP('2010-06-03 23:59:59', 'yyyy-MM-dd hh24:mi:ss')
GROUP BY
  COMPANY_T_1_."product_box"
LIMIT
  20000 OFFSET 0
SELECT
  COMPANY_T_1_."product_box" AS A_PRODUCT_2_,
  SUM(COMPANY_T_1_."profit_amt") AS A_PROFIT_3_
FROM
  "public"."company_sales_record" AS COMPANY_T_1_
WHERE
  COMPANY_T_1_."report_date" >= TO_TIMESTAMP('2010-05-29 00:00:00', 'yyyy-MM-dd hh24:mi:ss')
  AND COMPANY_T_1_."report_date" <= TO_TIMESTAMP('2010-05-31 23:59:59', 'yyyy-MM-dd hh24:mi:ss')
GROUP BY
  COMPANY_T_1_."product_box"
LIMIT
  20000 OFFSET 0
```

7.5.3.3.6. Filter field data

This topic describes how to filter field data by using the set filter feature.

Prerequisites

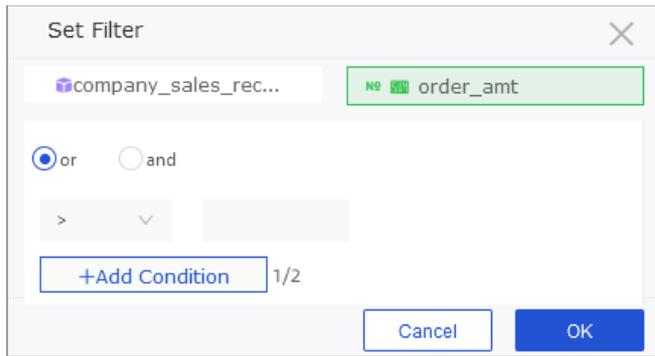
- The Quick BI service is purchased.
- A dataset is selected in the dataset selection area and is edited as need. For information about how to edit a dataset, see [Edit a dataset](#).

Context

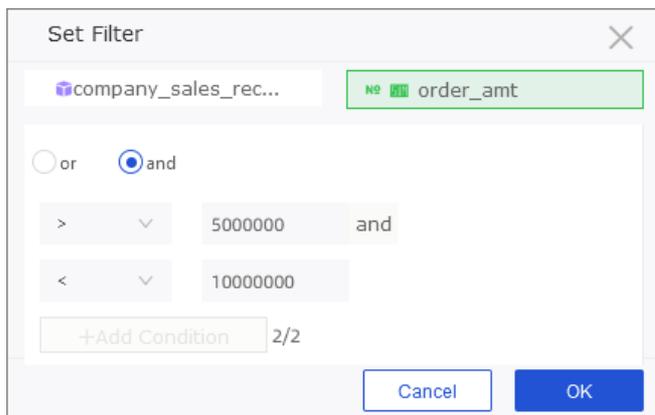
This topic describes how to filter data by using the set filter feature. The order_amt field is used in this example.

Procedure

1. [The required dashboard is displayed](#).
2. Drag order_amt to the filters field.
3. Click the  icon. The **Set Filter** dialog box appears.



4. In the Set Filter dialog box, configure a filter condition. For example, you can select Greater Than, Less Than, or Equal To and set the condition.



5. After you configure the parameters, click OK.
6. Click **Update**. The chart is updated.

7.5.3.3.7. Perform drill-down analysis

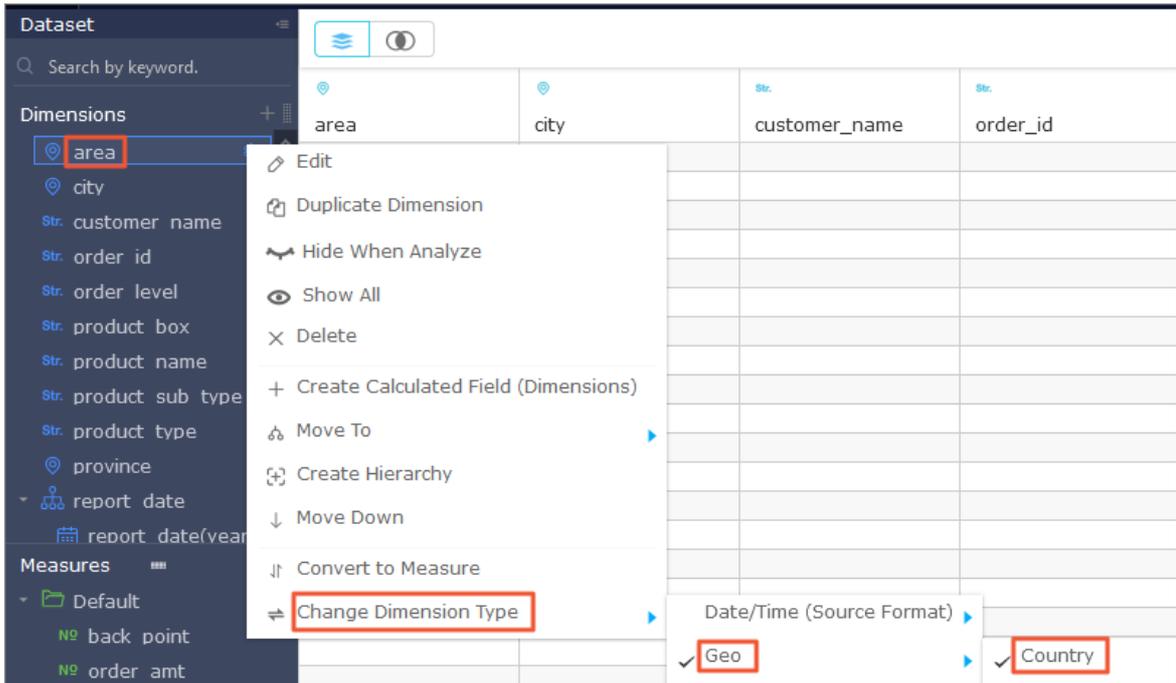
When you create a report, you need to analyze data at fine granularities to have a deep understanding of the data. Drill-down analysis can meet your requirements. Quick BI allows you to perform drill-down analysis. When you click an area in a chart, you can view the information of the provinces in this area, and when you click a province, you can view the information of the cities in this province. The following sections describe how to perform drill-down analysis. The company_sales_record dataset is used as an example.

Prerequisites

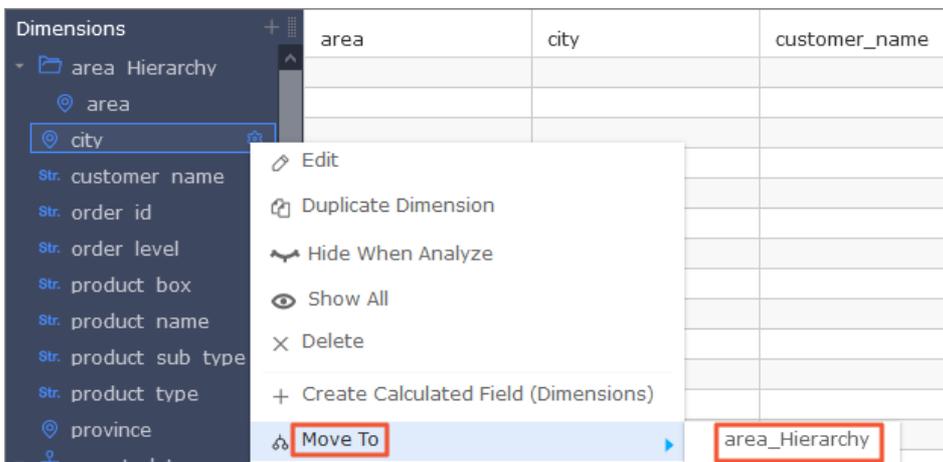
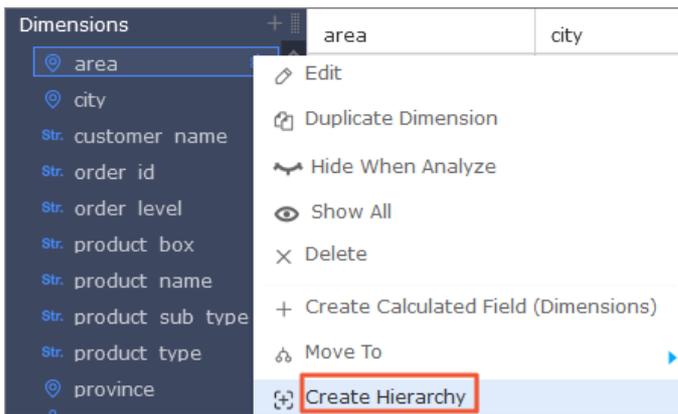
The Quick BI service is purchased.

Edit the dataset

1. [Go to the dashboard edit page.](#)
2. In the dataset selection section, find the required dataset and click the  icon.
3. Click the  icon next to the required field, choose **Change Dimension Type > Geo**, and then change area, province, and city to the required geographic fields, as shown in the following figure.



4. Create a hierarchy named **area_hierarchy** based on area and move province and city to this hierarchy, as shown in the following figure.



5. Save the dataset.

Create a dashboard to perform drill-down analysis

You can manually enable drill-down analysis on a dashboard.

1. In the dataset selection section, select the edited dataset.

For more information, see [Edit the dataset](#).

2. On the dashboard edit page, click the  icon.
3. On the Data tab, add the area and order_amt fields.
4. Click the  icon next to area.

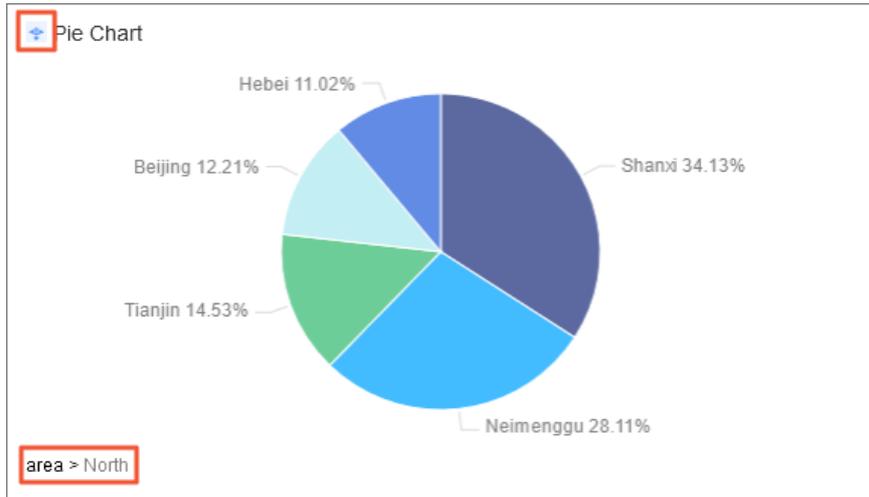


You can customize a drilling hierarchy in the **Drill/Dimension** section. If a hierarchy has been configured for the selected dimension, the hierarchy appears by default. You can also configure the hierarchy as required, as shown in the following figure.

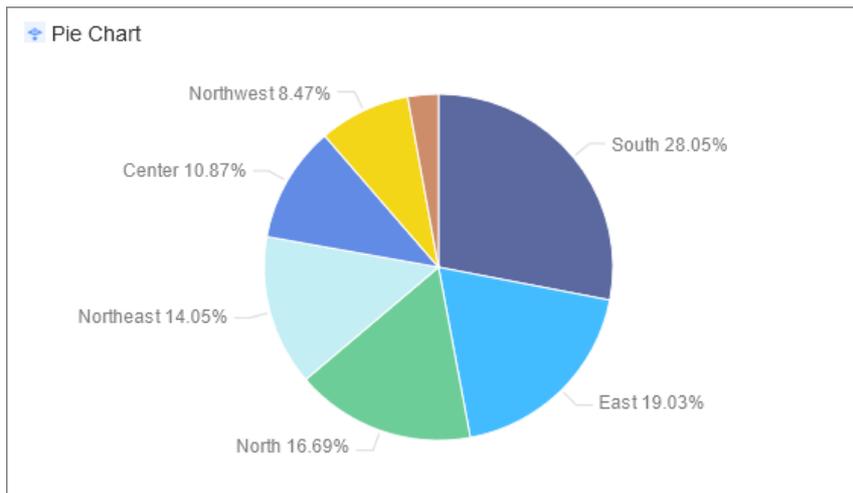


 **Note** The maximum number of drill-down levels is six.

After you configure the drilling hierarchy, the drill-down icon appears next to the chart title. When you click the North China area, information of the provinces in this area is displayed, and the drilling levels appear in the lower-left corner of the chart.



You can click the area in the lower-left corner to display the information of the area to which the current province belongs.



7.5.3.3.8. Filter interaction

You can use the filter interaction feature when you create multiple charts on a dashboard. This topic describes the filter interaction feature.

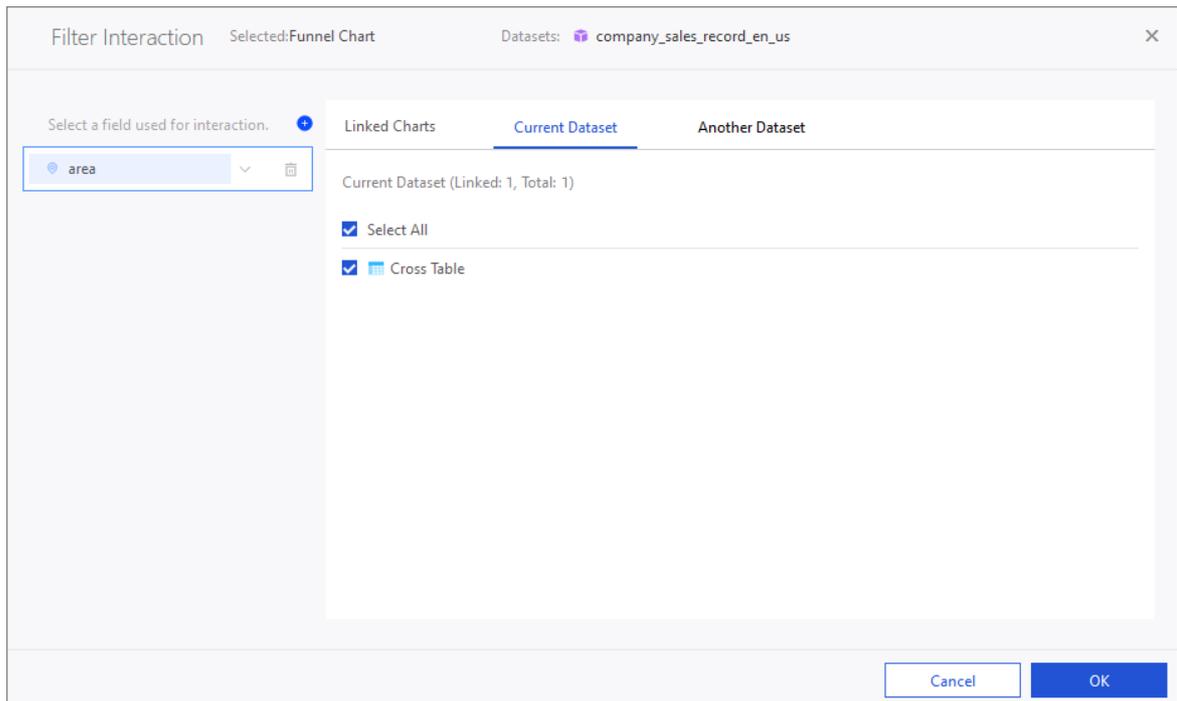
Prerequisites

- The Quick BI service is purchased.
- At least two charts are available in the display area of a dashboard.

Procedure

1. [Go to the dashboard edit page.](#)
2. Select a chart, such as a funnel chart.
3. In the Graphic Design area, click the **Advanced** tab.
4. On the **Advanced** tab, click the  icon next to Filter Interaction.

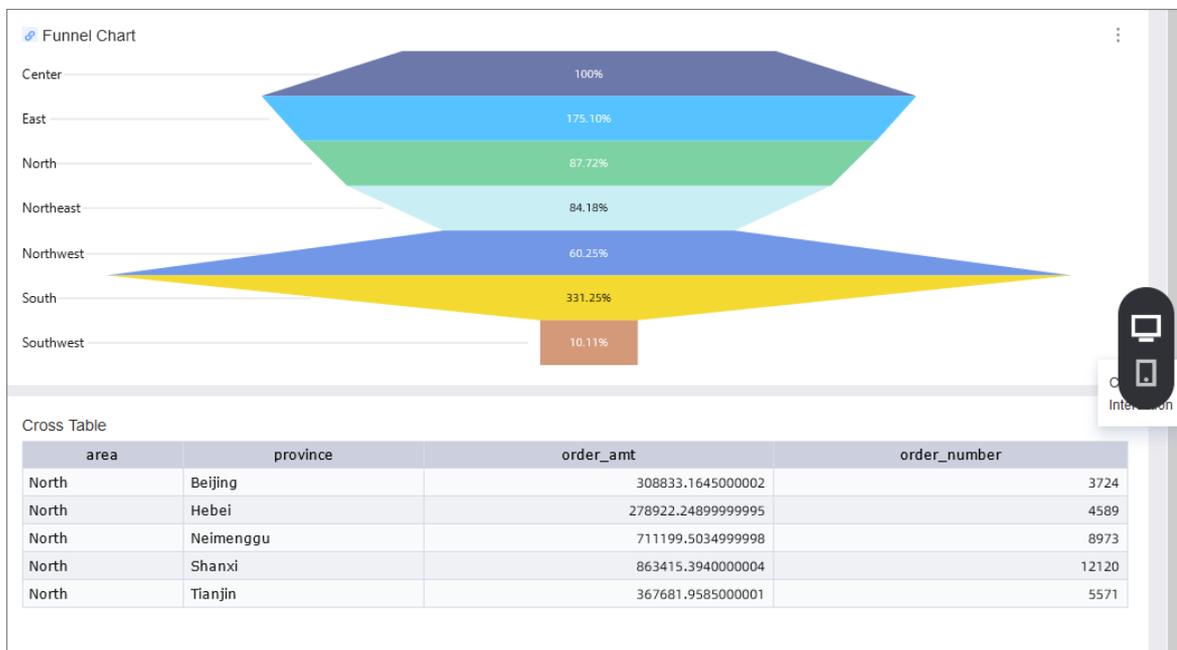
5. In the **Filter Interaction** dialog box, select a required dimension field and chart, and click **OK**.



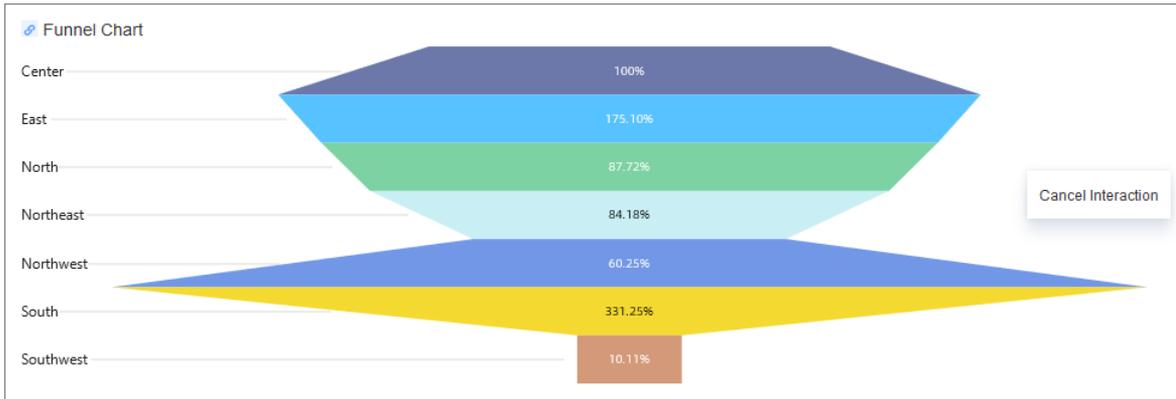
6. In the upper-right corner of the dashboard, click **Preview** to preview the current dashboard.



7. Click **China (Beijing)** in the funnel chart. The associated cross table displays data of the China (Beijing) region.



8. Move your pointer to any blank area in the funnel chart, and click **Cancel Interaction** to disable the filter interaction feature.



7.5.3.3.9. Metric analysis

This topic describes five methods of analysis: auxiliary line, trendline, prediction, anomaly detection and fluctuation analysis.

Prerequisites

- The Quick BI service is purchased.
- A dataset is selected in the dataset selection area and is edited as required.

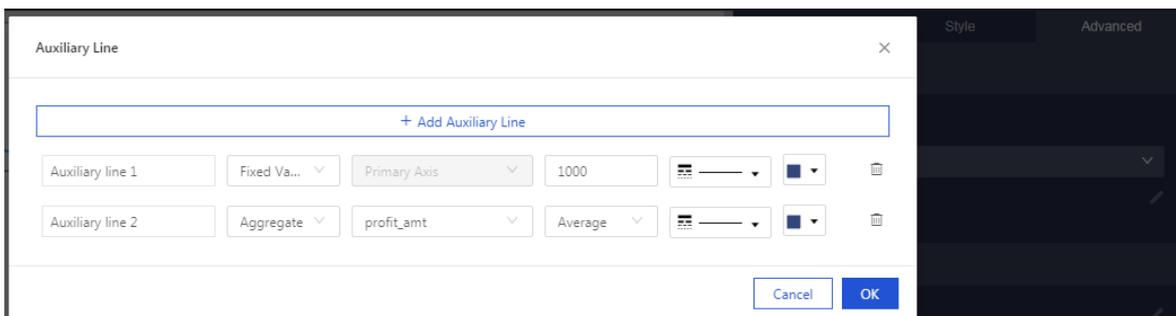
Background information

- Metric analysis allows you to analyze data from multiple perspectives. You can use this feature to obtain data trends and anomalies.
- Metric analysis supports five analysis methods: auxiliary line, trendline, prediction, anomaly detection, and fluctuation analysis.

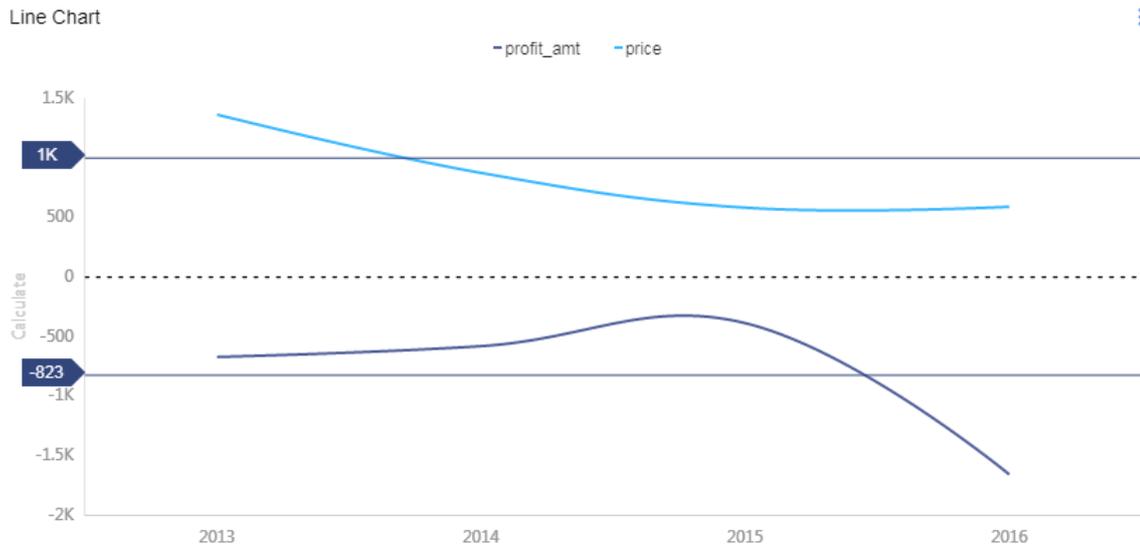
Auxiliary Line

You can add an auxiliary line to view the difference between the value of a measure and the value indicated by the auxiliary line. The value indicated by an auxiliary line can be a fixed or aggregate value. Aggregate values include average, maximum, minimum, and median values.

1. [Log on to the Quick BI console.](#)
2. On the Dashboards page of the Workspace, find the required dashboard and click the icon.
3. On the dashboard edit page, click the required chart.
4. On the **Advanced** tab of the **Graphic Design** page, click the icon next to **Gridlines** in the **Metric Analysis** section.
5. In the **Gridlines** section, click **Add Auxiliary Line** and configure the auxiliary line.



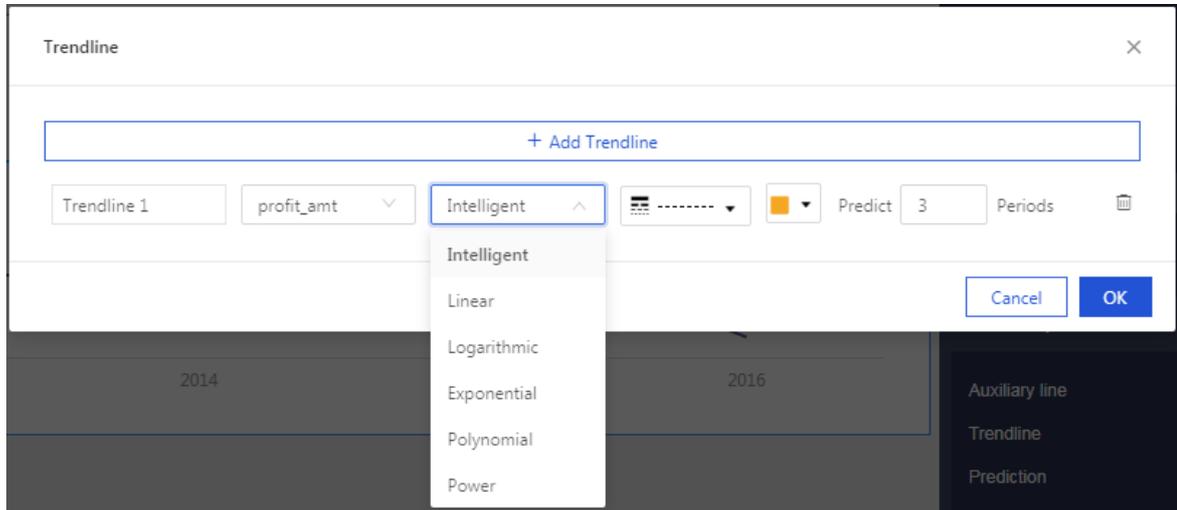
6. Click **OK**. The following figure shows a sample auxiliary line.



Trendline

A trendline displays the data trend. Types of trendlines include Intelligent, Linear, Logarithmic, Exponential, Polynomial, and Power.

1. On the **Advanced** tab of the **Graphic Design** page, click the  icon next to **Trendline** in the **Metric Analysis** section.
2. In the **Trendline** section, click **Add Trendline**. Select a measure, a trendline type, and the number of subsequent periods for which to predict trends.



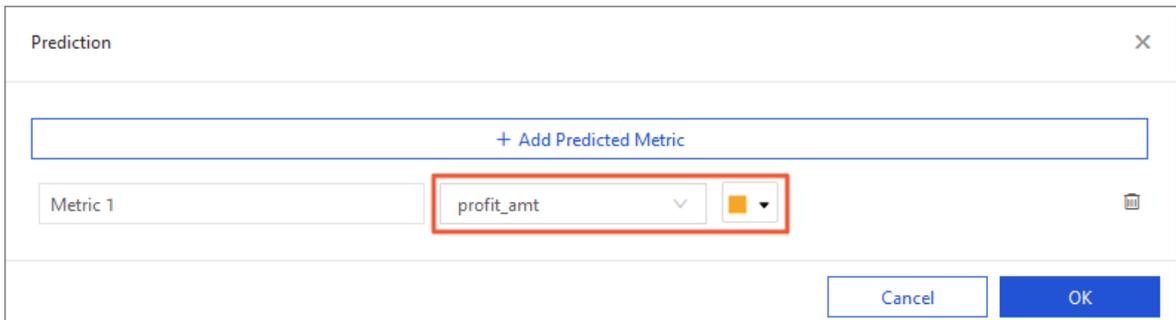
3. Click **OK**. The following figure shows a sample trendline.



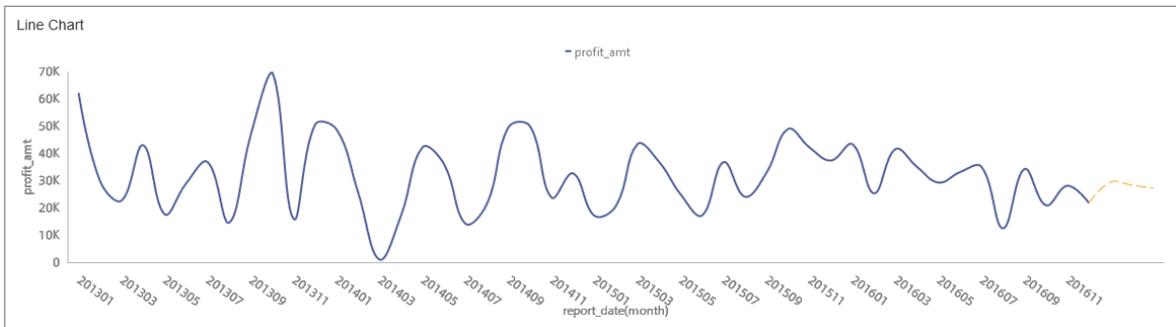
Prediction

You can add a predicted metric to view the trend of current data and predict future trends.

1. On the **Advanced** tab of the **Graphic Design** page, click the  icon next to **Prediction** in the **Metric Analysis** section.
2. In the **Prediction** section, click **Add Predicted Metric**. Select a measure and a color for the line.



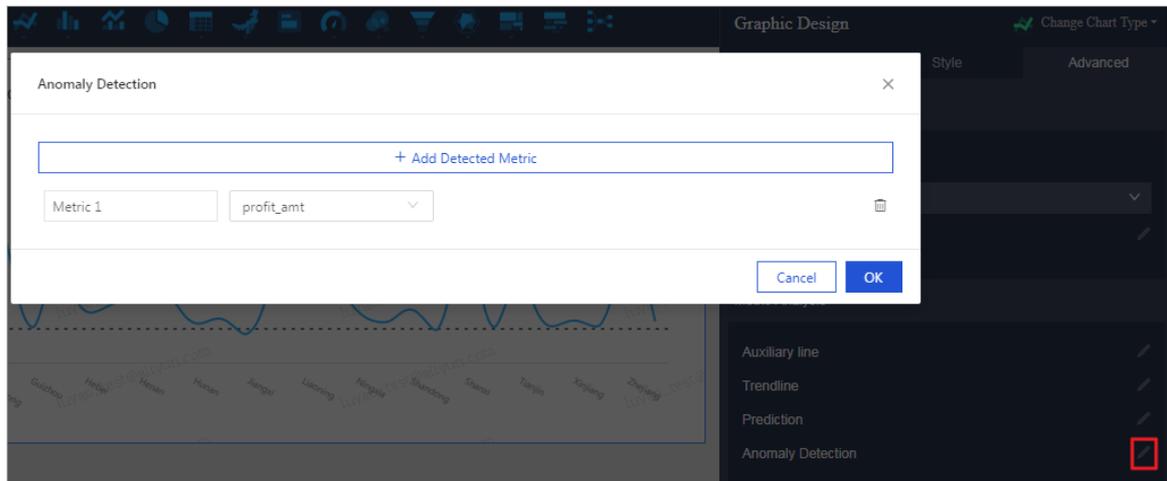
3. Click **OK**. The following figure shows sample prediction results.



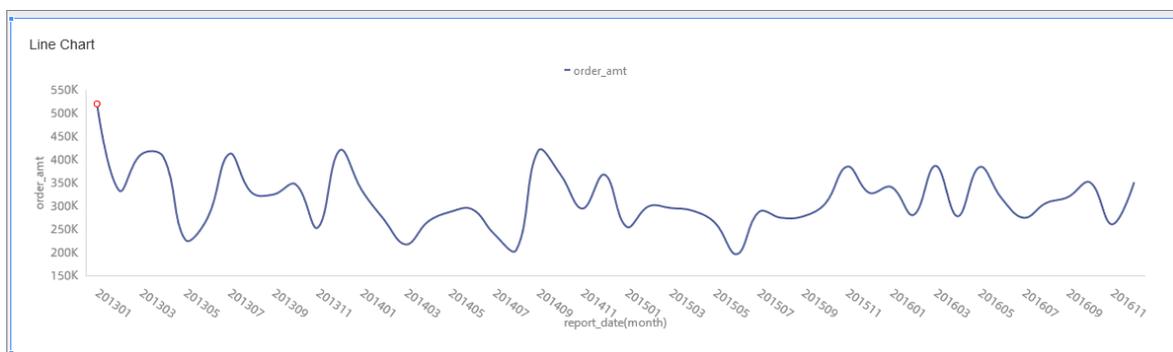
Anomaly Detection

You can use this feature to detect data anomalies.

1. On the **Advanced** tab of the **Graphic Design** page, click the  icon next to **Anomaly Detection** in the **Metric Analysis** section.
2. In the **Anomaly Detection** section, click **Add Detected Metric**. Select a measure.



3. Click **OK**. The following figure shows sample anomaly detection results.



Note An anomaly point in the line graph is marked with a red dot and an anomaly point in the column chart is marked with a full red column chart.

Fluctuation analysis

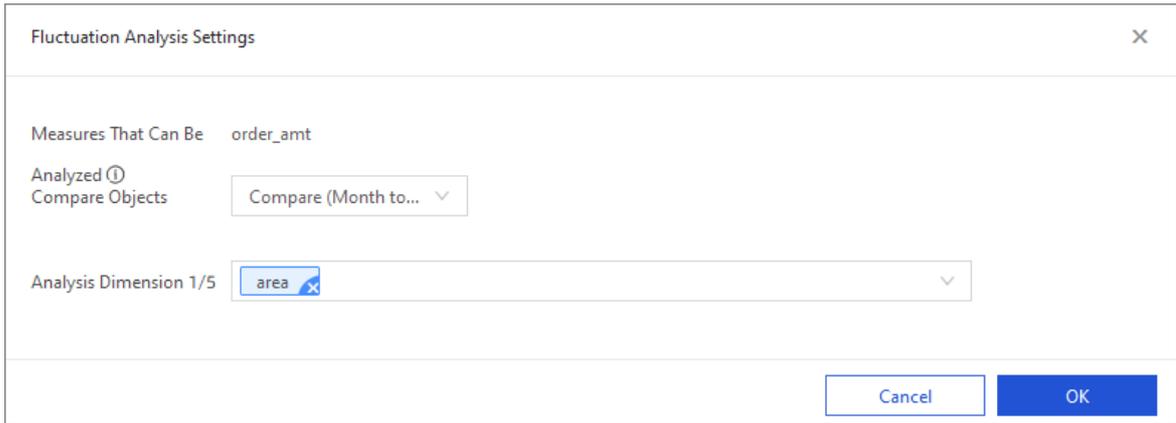
Fluctuation analysis uses machine learning algorithms to analyze the impact of dimensions on measures. You can view the contributions of dimensions to measure fluctuation and specific changes.

1. On the **Advanced** tab of the **Graphic Design** pane, click the  icon next to **Fluctuation Analysis** in the **Metric Analysis** section.

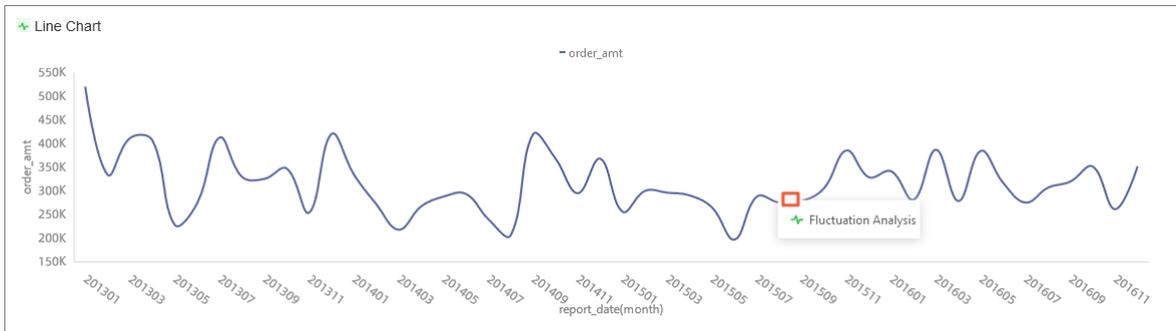
Note The charts that meet both of the following conditions support fluctuation analysis:

- A time dimension (except hour, minute, and second) is selected for Category Axis (Dim.).
- At least one measure uses the aggregate method SUM, COUNT, or COUNT DISTINCT.

2. On the **Fluctuation Analysis Settings** page, select a comparison object and an analysis dimension, and click **OK**.



3. Click a time point in the chart and then click **Fluctuation Analysis**.



4. In the **Fluctuation Analysis** section, select a measure to view the key factors that affect the measure.

Fluctuation Analysis 201508 Measures That Can Be Analyzed: order_amt

201508, order_amt: 275.21K; Compare with 201507 (order_amt: 287.91K), Change -12.69K, **-4.41%**.

Key Influencing Factors

areaisSouth Change contribution to order_amt is **490%**, order_amt: 51.79K; Change -62.15K, **-54.5%**;

areaisNorth Change contribution to order_amt is **284%**, order_amt: 37.29K; Change -36.01K, **-49.1%**;

areaisCenter Change contribution to order_amt is **34.2%**, order_amt: 19.69K; Change -4341.29, **-18.1%**;

Reverse Influencing Factors

areaisNortheast Change contribution to order_amt is **-327%**, order_amt: 64.11K; Change 41.53K, **184%**;

areaisNorthwest Change contribution to order_amt is **-206%**, order_amt: 33.06K; Change 26.20K, **382%**;

areaisSouthwest Change contribution to order_amt is **-94.9%**, order_amt: 12.86K; Change 12.05K, **1491%**;

areaisEast Change contribution to order_amt is **-79.0%**, order_amt: 56.41K; Change 10.03K, **21.6%**;

⚙️ Change contribution of a dimension = Change in the dimension / Total change

OK

7.5.3.4. Dashboard display area

7.5.3.4.1. Overview

This topic describes the features of the dashboard display area.

In the display area of a dashboard, you can perform the following operations on one or more charts:

- Adjust chart positions
- View chart data
- Change chart types
- Add to favorites
- Delete a chart

7.5.3.4.2. Toolbar

This topic describes the features of the toolbar.

The toolbar of a dashboard allows you to save, preview, and edit the dashboard, as shown in [Dashboard toolbar](#).

Dashboard toolbar



7.5.3.4.3. Adjust chart positions

Multiple charts may be displayed on the same dashboard. In this case, you can drag the charts to adjust their positions.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Go to the target dashboard](#).
2. Select a chart or widget.
3. Drag the chart or widget to the desired position.

 **Note** You can drag a chart or widget to anywhere within the display area of the dashboard.

7.5.3.4.4. View chart data

This topic describes how to view chart data.

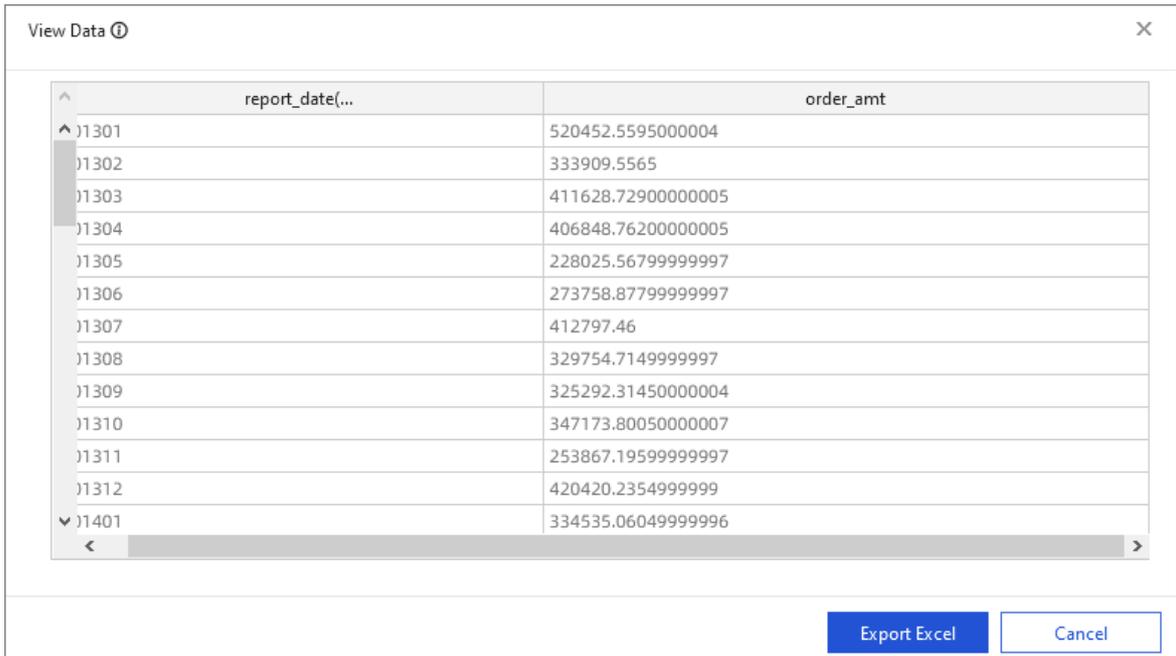
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Go to the dashboard edit page](#).
2. In the dashboard display section, select a chart, such as a funnel chart.
3. Move the pointer over the upper-right corner of the chart, click the  icon, and then select **View Data**.

4. On the page that appears, click **Export Excel** to download data to your on-premises machine.



7.5.3.4.5. Change chart types

You can select different chart types from the toolbar on the top of the dashboard.

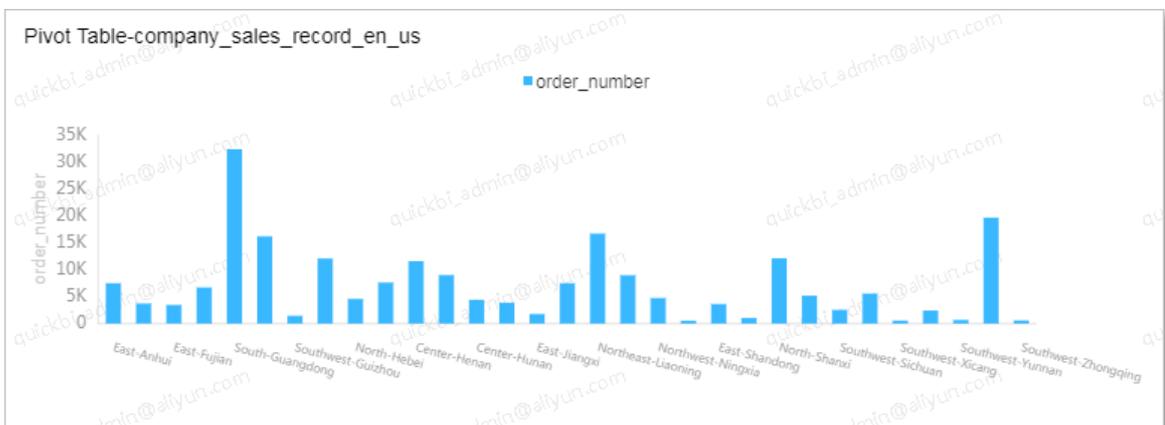
Prerequisites

The Quick BI service is purchased.

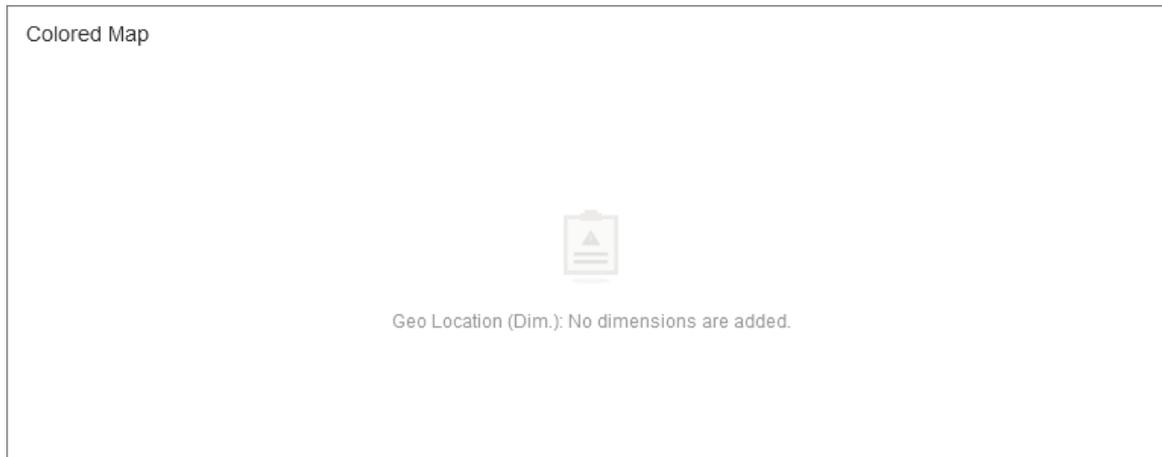
Procedure

1. Go to the dashboard edit page.
2. Select a chart, such as a cross table.
3. In the Graphic Design area, click **Change Chart Type** and select another chart type, such as a vertical bar chart.
4. Click the **Vertical Bar Chart** icon to change the chart type.

The system then converts the cross table to a vertical bar chart.



If the switch between chart types fails, the elements of the selected chart type do not match those of the current chart type. You need to modify the problematic data fields based on the requirements of the selected chart type.



To change the chart type, follow the instructions to manually adjust the dimensions and measures.

7.5.3.4.6. Add to Favorites

This topic describes how to add a dashboard to the Favorites tab.

Prerequisites

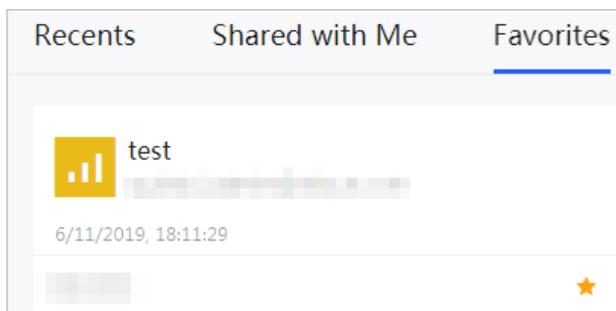
The Quick BI service is purchased.

Procedure

1. The required dashboard is displayed.
2. On the display area of the dashboard, click the  icon in the top navigation bar.



3. On the Quick BI homepage, you can click the **Favorites** tab to view the dashboards that you have added.



7.5.3.4.7. Delete a chart

This topic describes how to delete a chart.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Access a dashboard](#).
2. Select a chart, such as a radar chart.
3. Move the pointer over the upper-right corner of the chart and click the  icon. Select **Delete** to delete the chart.

7.5.3.4.8. Widgets

7.5.3.4.8.1. Overview

This topic describes the widgets of a dashboard.

The display section of a dashboard provides the following widgets:

- Query control
- Text area
- iFrame
- Tab
- Image

7.5.3.4.8.2. Filter bar

In a dashboard, you can use the Filter Bar widget to query data in one or more charts.

Create a filter bar

This topic describes how to create a filter bar.

Prerequisites

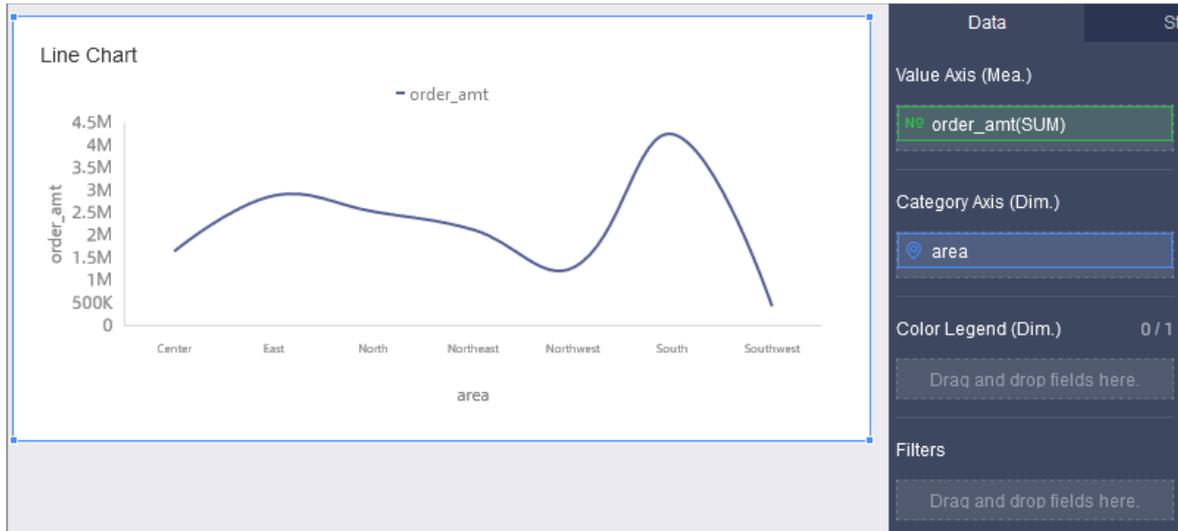
The Quick BI service is purchased.

Context

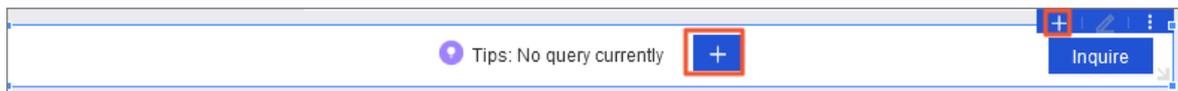
Filter bars allow you to associate charts from the same dataset or those from different datasets.

Procedure

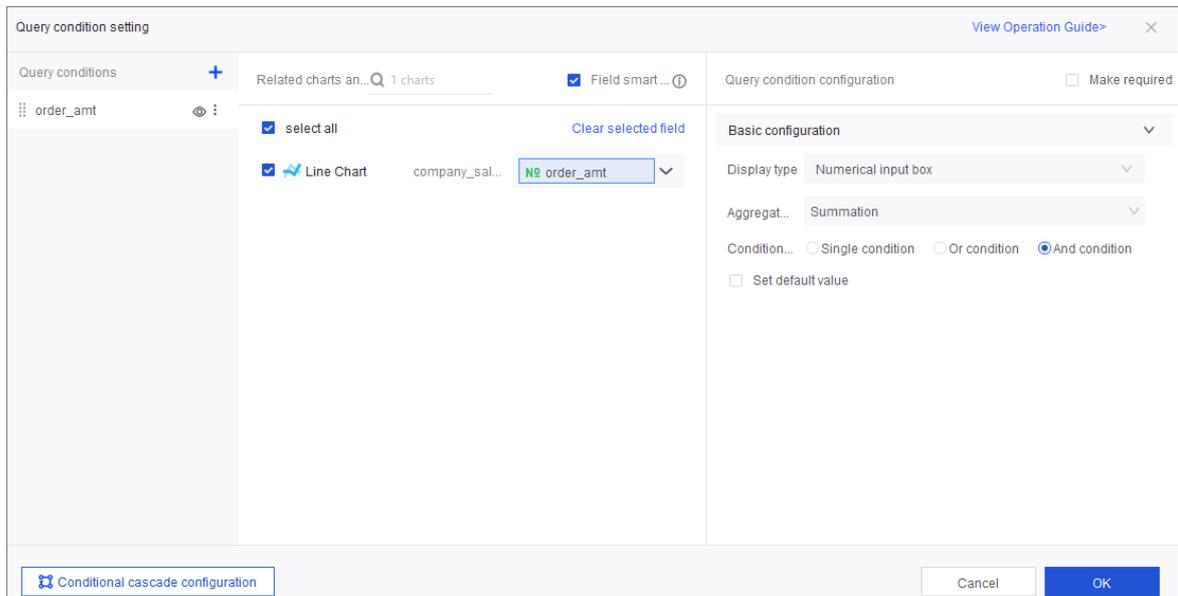
1. [Go to the dashboard edit page](#).
2. Create a line chart on the dashboard edit page. For more information, see [Create a line chart](#).



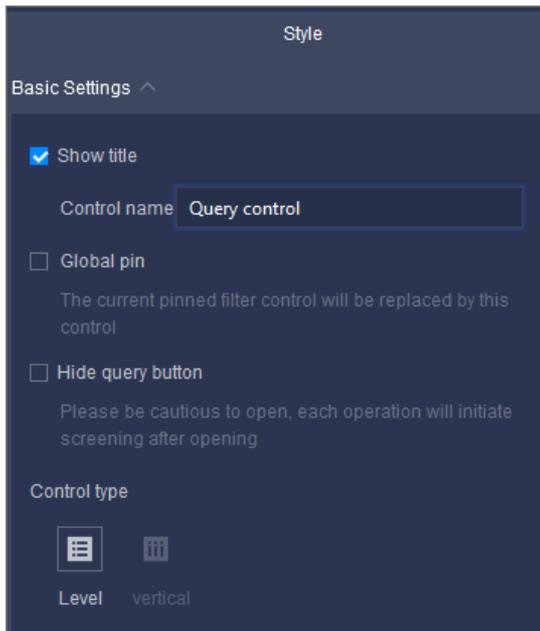
3. In the menu bar, click the  icon to add a filter bar.
4. In the dashboard display section, find the filter bar and click the  icon.



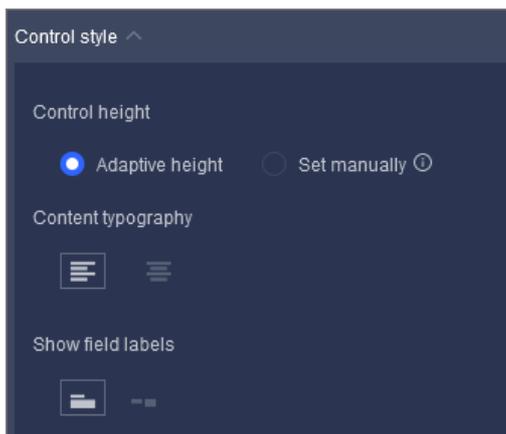
5. In the Set Query Conditions dialog box, configure the filter conditions and click OK.



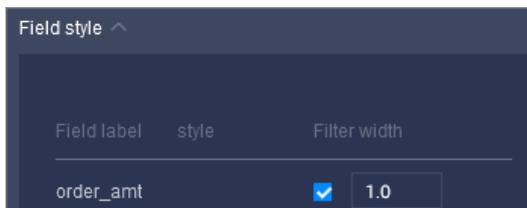
6. On the Style tab, configure the style of the filter bar.
 - o In the **Basic Settings** section, configure Show Title, Global Pin, Hide Query Button, and Control Type.



- o Control Style: In this section, you can configure Control height, Content typography, and field label positions.



- o Field Style: In this section, you can configure the styles and widths of fields.



7. In the filter bar, configure filter conditions and click **Query**. The chart that is associated with the field is automatically updated.



Perform cascade filtering

The Filter Bar widget supports cascade filtering, which makes it easier for you to specify multiple filter conditions.

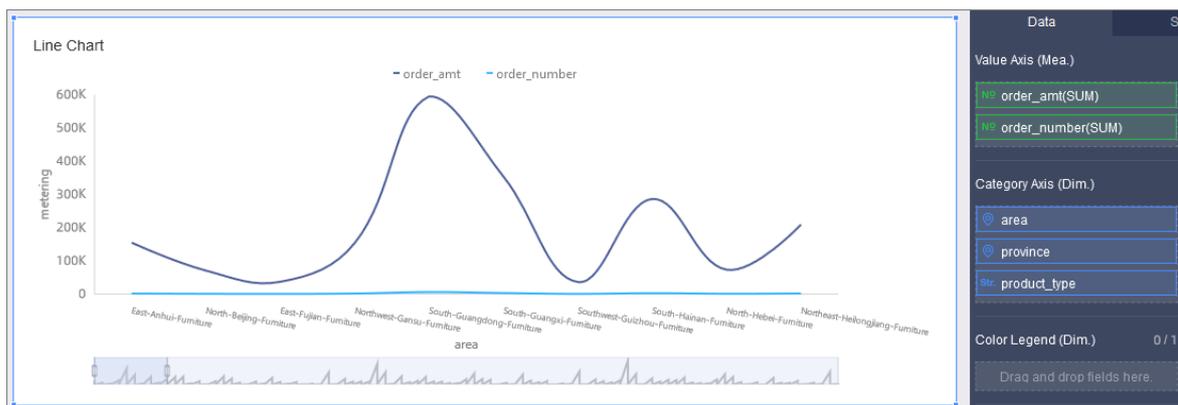
Prerequisites

- The Quick BI service is purchased.
- Two filter bars are created on the dashboard, with Display Mode set to Drop-Down List and Source set to Single Dataset. For more information, see [Create a filter bar](#).

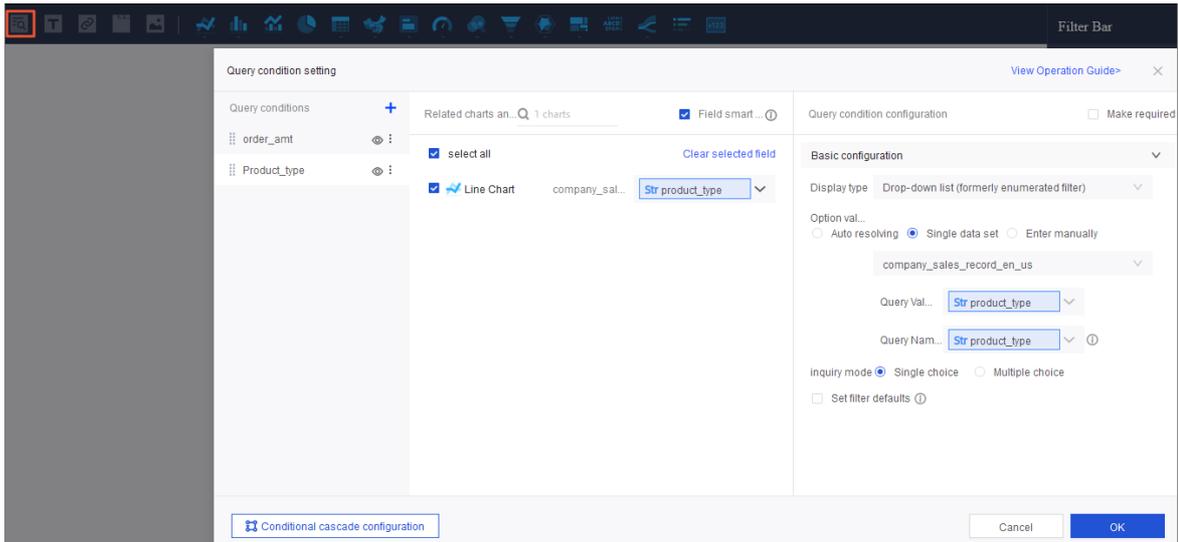
Procedure

1. [Go to the dashboard edit page](#).
2. Create a line chart on the dashboard edit page.

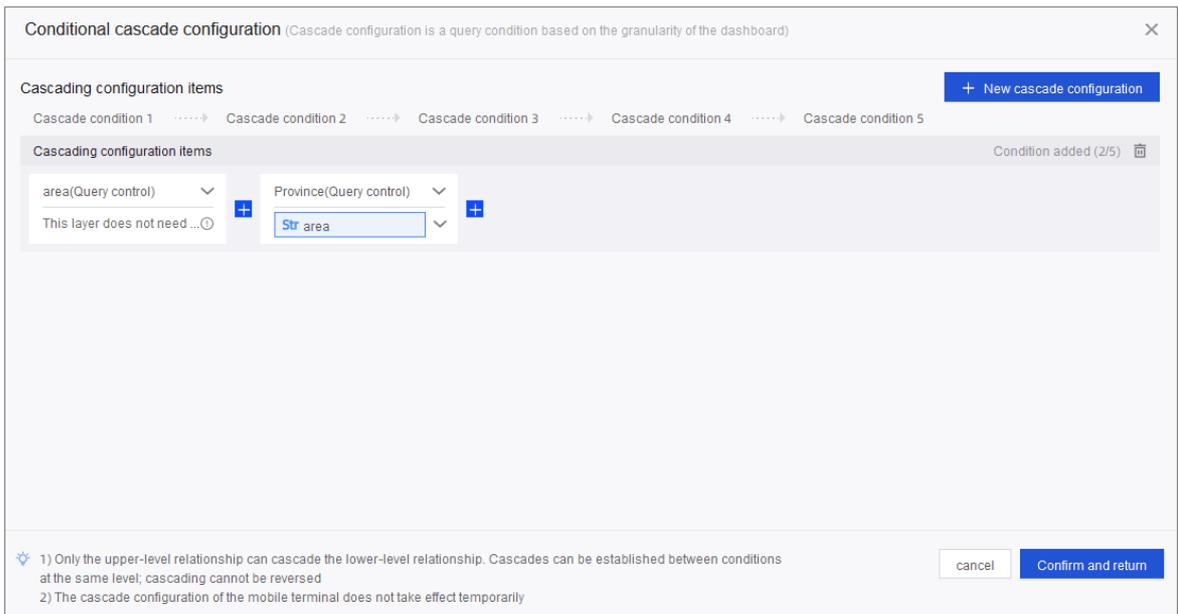
For more information, see [Create a line chart](#).



3. Add and configure the filter bar, and then click **OK** to save the configuration.



4. In the **Set Query Conditions** dialog box, click **Conditional Cascade Configuration**.
5. In the **Cascade Configuration Items** section in the **Conditional Cascade Configuration** dialog box, select **area** and click the **+** icon next to the field.
6. Select the conditional field and associated field.



- To associate multiple conditions, click **Add Cascade Configuration Item** in the upper-right corner.
7. After the configuration is completed, click **Confirm and Return**.
 8. Click **Query** in the filter bar.



Note If you want to view details of the cascade, move the pointer over the cascade icon in the upper-right corner of the line chart.

Use SQL statement parameters in the Filter Bar widget

This topic describes how to use parameters in SQL statements for ad hoc analysis as the filter fields in the Filter Bar widget.

Prerequisites

Parameters are set in SQL statements for ad hoc analysis.

Procedure

1. Click the Filter Bar icon on the dashboard edit page. On the Data tab, select a dataset. If the dataset contains parameters of SQL statements, the parameters appear at the lower part of this tab and are displayed in orange.

Note Parameters in SQL statements do not support cascade filtering.

2. Select a parameter, click the **Set Filter** icon. On the Set Filter dialog box that appears, perform filter settings.

If you select **Filter by Condition** for parameters of the date, numeric, and text types, see [Query data by date](#), [Query numeric data](#), and [Filter bars](#).

If you select **Filter by Value** for parameters of the string type, set the enumeration value range and then enter the parameter values.

Select the parameter that you want to use for data filtering and click Query.

Query data of a specific time range

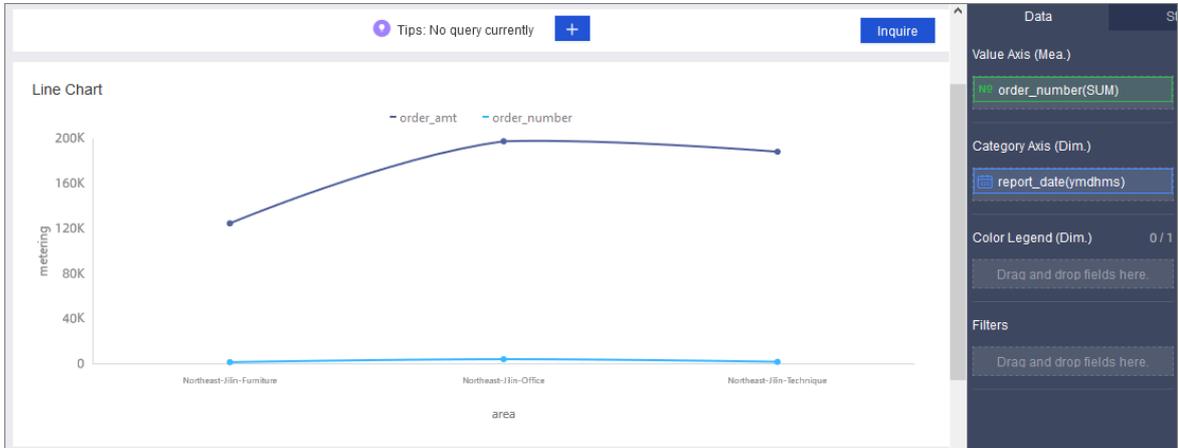
The Filter Bar widget allows you to query data of a specific time range. You can set a time range, manually set default values, and customize shortcuts to query time-related data.

Prerequisites

The Quick BI service is purchased.

Procedure

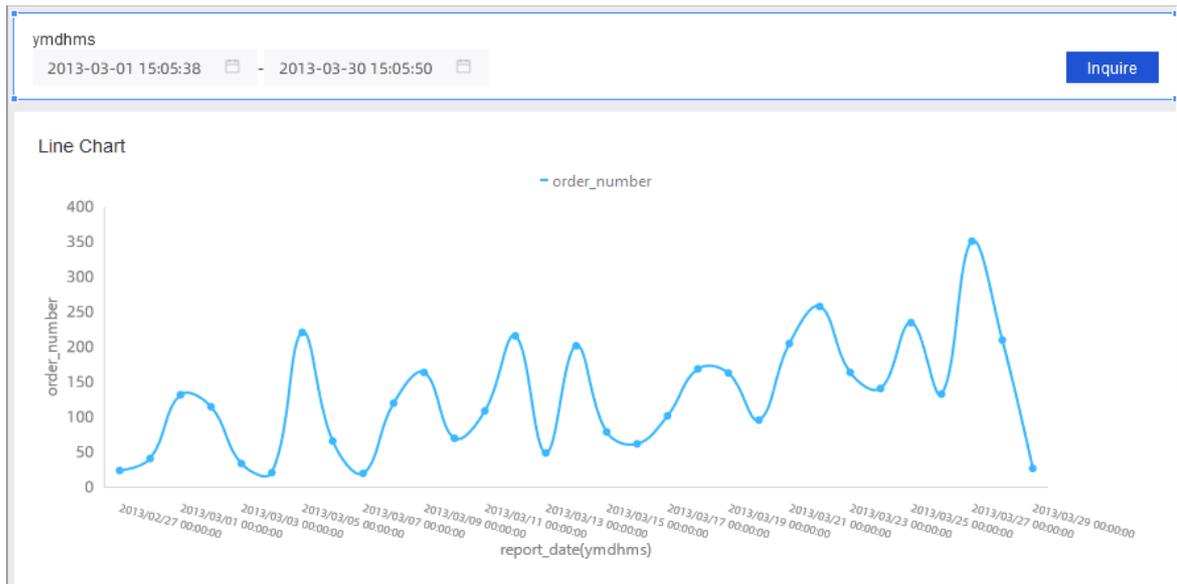
1. The required dashboard is displayed.
2. Create a line chart and a filter bar on the dashboard edit page.



3. In the Set Query Conditions dialog box, configure query conditions. In this example, configure YYYY/MM/DD HH:mm:ss.

The screenshot shows the "Query condition setting" dialog box. It has a title bar with "View Operation Guide" and a close button. The main area is divided into several sections: "Query conditions" with a "+" icon and a list of conditions including "ymdhms"; "Related charts and..." with "2 charts" and a "Field smart..." checkbox; "Query condition configuration" with a "Make required" checkbox. Under "Basic configuration", there are settings for "Display type" (Date selection), "Time granul" (Year-month-day-hour-minute-second), "Display form" (YYYY-MM-DD HH:mm:ss), "Set time filter range" (Not configured), "Screening m" (singlesecond, secondInterval), and "Interval type" (Start at, Ends at, Time interval). At the bottom, there is a "Conditional cascade configuration" button, "Cancel", and "OK" buttons.

4. After the parameters are configured, click **OK**.
5. Specify a time duration and click **Query**.



Query text data

This topic describes how to use the Query control widget to query text data.

Prerequisites

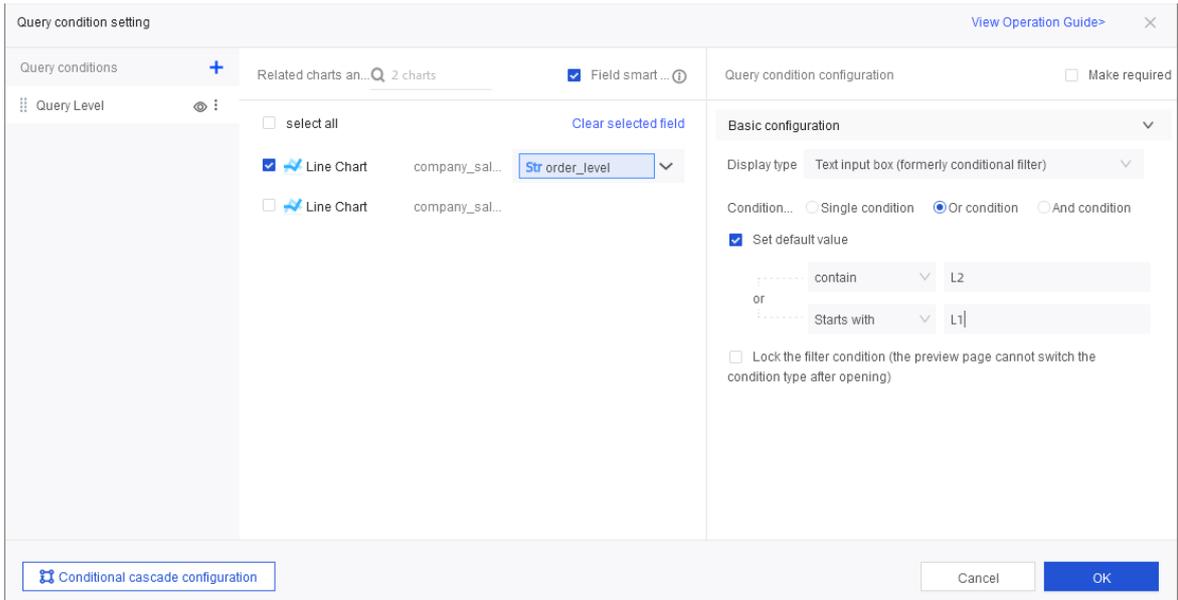
The Quick BI service is purchased.

Procedure

1. The required dashboard is displayed.
2. Create a line chart and a query control on the dashboard edit page.



3. In the Set Query Conditions dialog box, configure query conditions. In this example, order_level is selected.



- 4. After the parameters are configured, click **OK**.
- 5. In the query control, click **Query**



Query numeric data

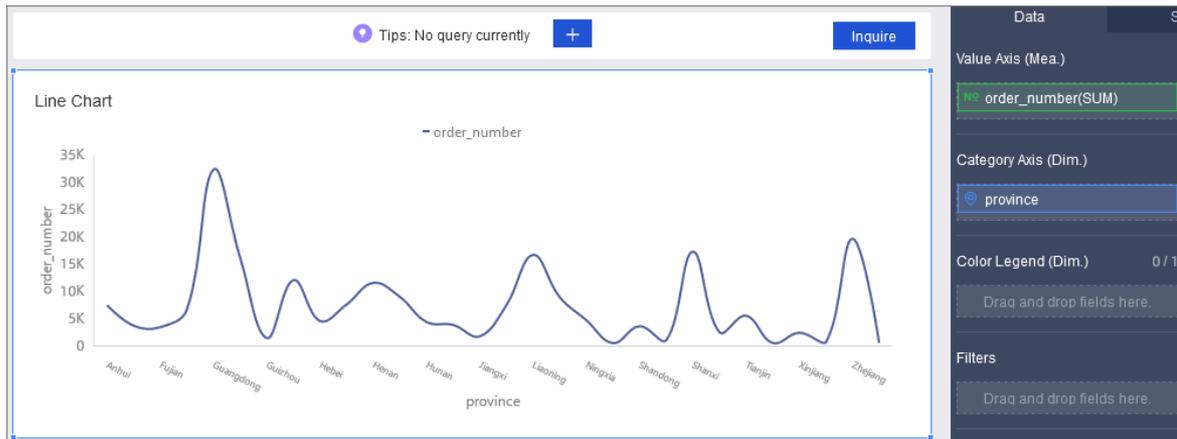
This topic describes how to use the Filter Bar widget to query numeric data.

Prerequisites

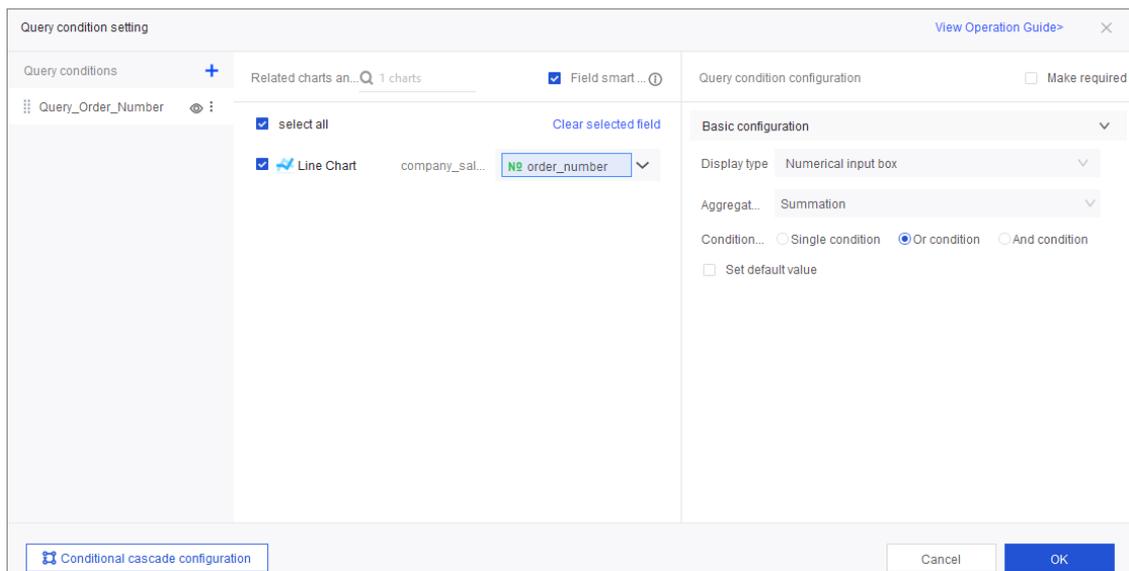
The Quick BI service is purchased.

Procedure

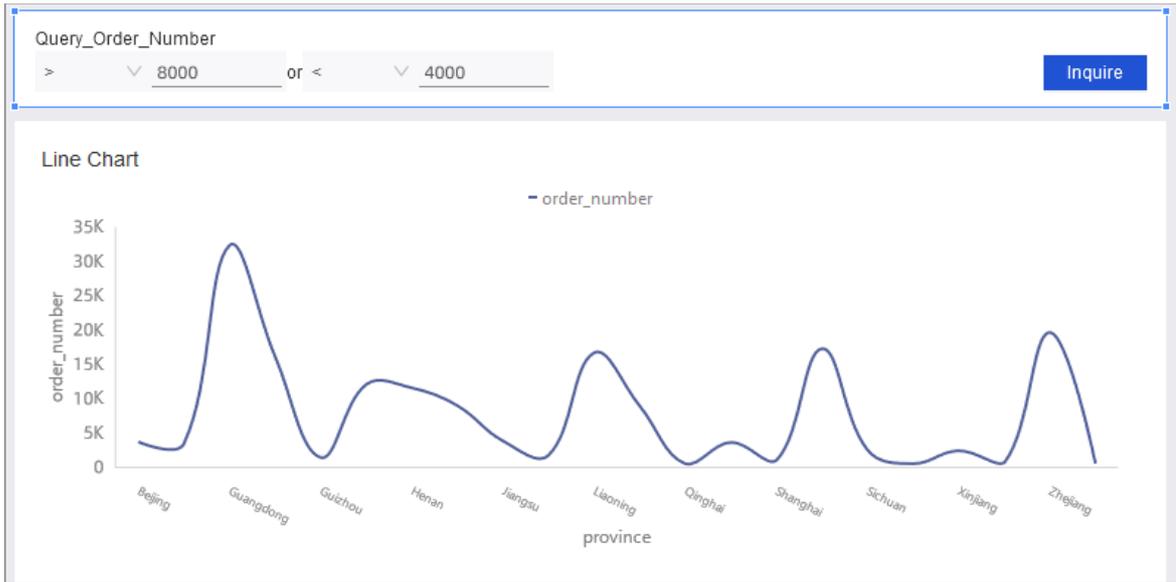
1. [Go to the dashboard edit page.](#)
2. Create a line chart and a filter bar on the dashboard edit page.



3. In the Set Query Conditions dialog box, configure query conditions.
 - i. Set the query condition name to **Order quantity query**.
 - ii. Select Line chart and the **order_number** field and associate them with the query control.
 - iii. When the field type is numeric, set Aggregation method and **Conditional form** as required. **Display type** is automatically set to **Numerical input box**. Supported aggregation methods include multiple aggregate functions, variance, and standard deviation. Supported conditional forms include **Single condition**, **Or condition** or **And condition**.



4. After the parameters are configured, click **OK**.
5. In the filter bar, configure filter conditions and click **Query**.



7.5.3.4.8.3. Compound Query Control

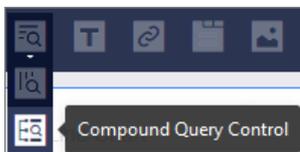
Add filter conditions

Prerequisites

- The Quick BI service is purchased.
- The target dashboard is displayed.

Procedure

1. Click the **Compound Query Control** icon.

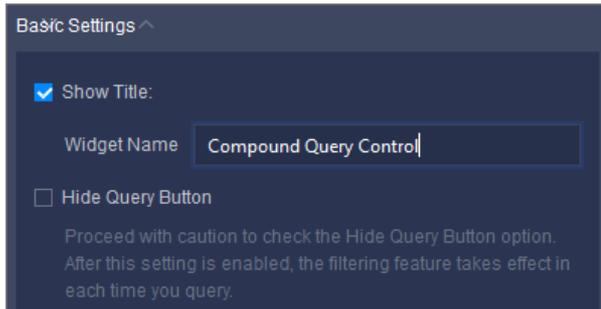


2. Click the **Data** tab and perform the following configurations:
 - Select a dataset from the Dataset drop-down list.
 - Click  to go to the dataset edit page to edit dimensions and measures as required.

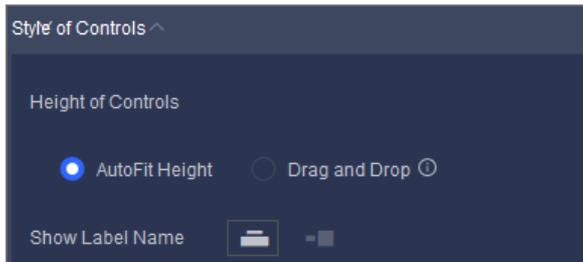
3. Click **Configure Selector** and add fields for query.



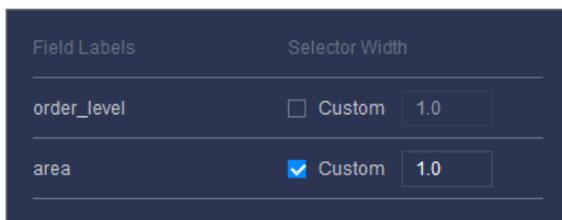
4. Click the **Style** tab and perform the following configurations:
 - In the **Basic Settings** section, configure whether to show the widget title and query button.



- o In the **Style of Controls** section, set the widget height and label name position.

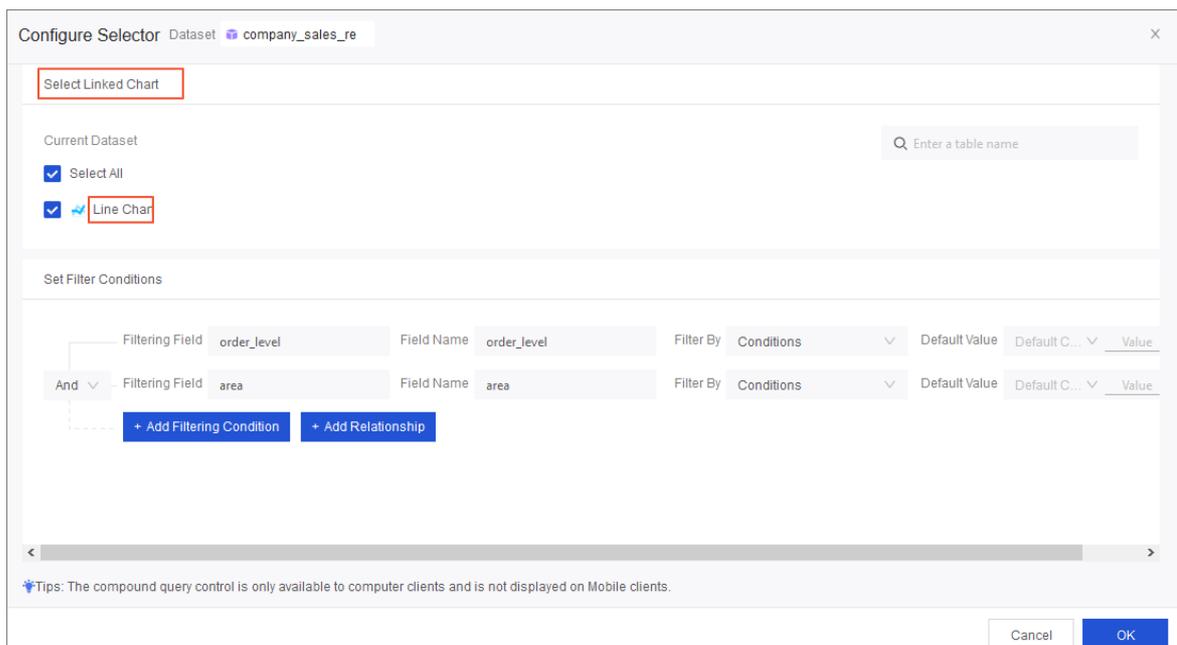


- o In the **Style of Field** section, set the field width.



5. Click **Configure Selector**. In the Configure Selector dialog box that appears, select charts for filter interaction.

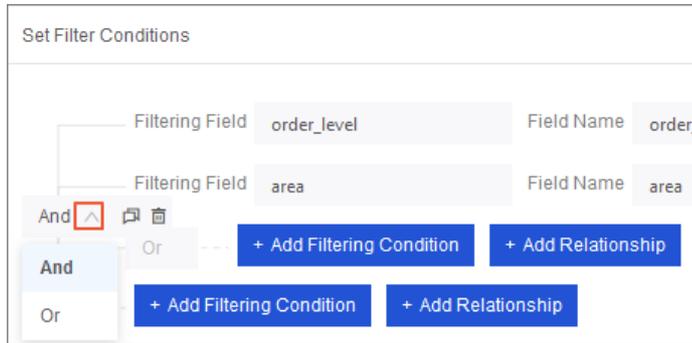
Note Compound Query Control supports only filter interaction of charts from the same dataset. The widget does not support cascade filtering.



6. In the **Set Filter Conditions** section, select filter fields and perform the following configurations:
 - i. Click **Add Filtering Condition** to add a filter condition under the current layer.
 - Search for or select a field from the Filtering Field drop-down list, change Field Name, specify Filter By, and set Default Value.
 - If you want to delete a filter condition, click the Delete icon next to the filter condition.
 - ii. Click **Add Relationship** to add a relationship node in the current layer.



- Click the outer-layer relationship node and select a relationship type as required.



- If the outer-layer relationship is **And**, the default inner-layer relationship is **Or**. If the outer-layer relationship is **Or**, the default inner-layer relationship is **And**. The relationship types of adjacent layers are opposite. They can be **And-Or**, **Or-And**, **And-Or-And**, or **Or-And-Or**.
- Click the Delete icon next to a node to delete all filter fields and relationships under the node.



Note You can add a maximum of three relationships for each node.

7. Click **OK** to complete the configuration of the Compound Query Control widget.

Query data of a specific time range

This topic describes how to use the Compound Query Control widget to query data of a specific time range. Data of a specific time range can be filtered by using Short cut or Filter By Date.

Prerequisites

- The Quick BI service is purchased.
- The target dashboard is displayed.

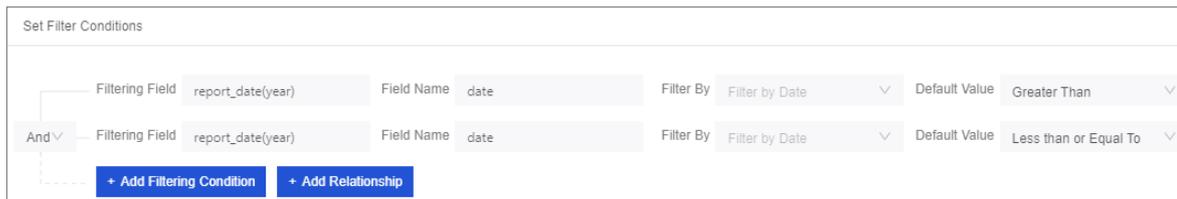
Procedure

1. Click the **Compound Query Control** icon. On the Data tab, select a dataset.
2. Click **Configure Selector**. In the Configure Selector dialog box that appears, select charts for filter interaction.

3. Click **Add Filtering Condition** to set filter conditions.

Note If you filter data of a certain period of time, you must add a date field first, and set **Default Value** to **Greater Than** or **Greater than or Equal To**. Then, add the same date field, and set **Default Value** to **Less than or Equal To** or **Less Than**.

- o Click **Add Filtering Condition**. Set Filtering Field to **report_date(year)**, Field Name to **report_date(year)**, and Default Value to **Greater Than, 2010**.
- o Click **Add Filtering Condition**. Set Filtering Field to **report_date(year)**, Field Name to **report_date(year)**, and Default Value to **Less than or Equal To, 2011**.



Note You can use the **Shortcut** feature to customize time ranges only for fields of day and month types.

4. Click **OK**.
5. Click **Query** in the display area of the Compound Query Control widget. All the charts that are filtered by the widget are updated.



Query text data

This topic describes how to use the Compound Query Control widget to query text data. Text data can be filtered in three methods: Enum - Single Choice, Enum - Multiple Choice, and Conditions.

Prerequisites

- The Quick BI service is purchased.
- **The target dashboard is displayed.**

Procedure

1. Click the **Compound Query Control** icon. On the Data tab, select a dataset.
2. Click **Configure Selector**. In the Configure Selector dialog box that appears, select charts for filter interaction.
3. In the **Set Filter Conditions** section, set filter conditions.
 - i. Click **Add Filtering Condition**. Then, set Filtering Field to **area**, Field Name to **area**, Filter By to **Enum - Single Choice**, and Default Value to **North**.
 - ii. Click **Add Filtering Condition**. Then, set Filtering Field to **order_level**, Field Name to **order_level**, Filter By to **Conditions**, and Default Value to **Matches, L1**.
4. Click **OK**.
5. Click **Query** in the display area of the Compound Query Control widget. All the charts filtered by the

widget are updated.

Query numeric data

This topic describes how to use the Compound Query Control widget to query numeric data.

Prerequisites

- The Quick BI service is purchased.
- **The target dashboard is displayed.**

Procedure

1. Click the **Compound Query Control** icon. On the Data tab, select a dataset.
2. Click **Configure Selector**. In the Configure Selector dialog box that appears, select charts for filter interaction.
3. In the **Set Filter Conditions** section, set filter conditions.
 - i. Click **Add Filtering Condition**. Then, set Filtering Field to **order_number**, Field Name to **order_number**, and Default Value to **Greater Than, 30**.
 - ii. Click **Add Filtering Condition**. Then, set Filtering Field to **order_amt**, Field Name to **order_amt**, and Default Value to **Greater Than, 10000**.
4. Click **OK**.
5. Click **Query** in the display area of the Compound Query Control widget. All the charts filtered by the widget are updated.

Note Only numeric values can be filtered. The data is displayed based on the aggregation method selected for measures in a chart. Assume that the aggregation method of the order_number field in a cross table is SUM. If you use the Compound Query Control widget to filter data records with order quantities greater than 30, the cross table displays all data records whose order quantity is greater than 30 and aggregation method is SUM.

7.5.3.4.8.4. Text Area

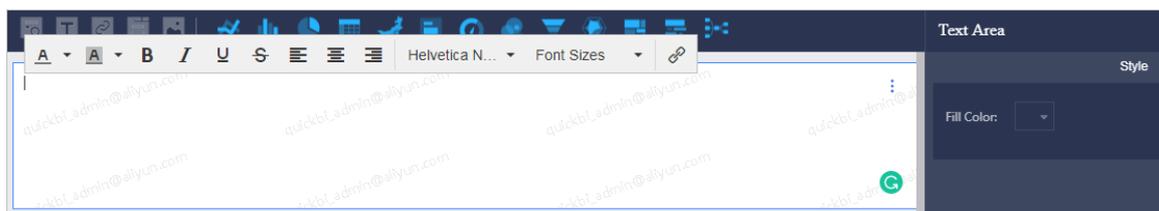
The Text Area widget allows you to enter text into a text area, for example, a title for a chart.

Prerequisites

The Quick BI service is purchased.

Procedure

1. **Go to the target dashboard.**
2. Click the **Text Area** icon.
3. Enter text in the text box.



7.5.3.4.8.5. IFrame

You can use the IFrame widget to insert web pages to query Internet data or browse web pages or websites related to the data on the current dashboard in real time.

Prerequisites

The Quick BI service is purchased.

Procedure

1. Go to the target dashboard.
2. Click the IFrame icon.
3. In the URL input box, enter the address of the web page you want to visit.



Note The web page address must start with https.

If you want to delete the current IFrame widget, click the **More** icon in the upper-right corner of the IFrame widget and select **Delete**.

7.5.3.4.8.6. Tab

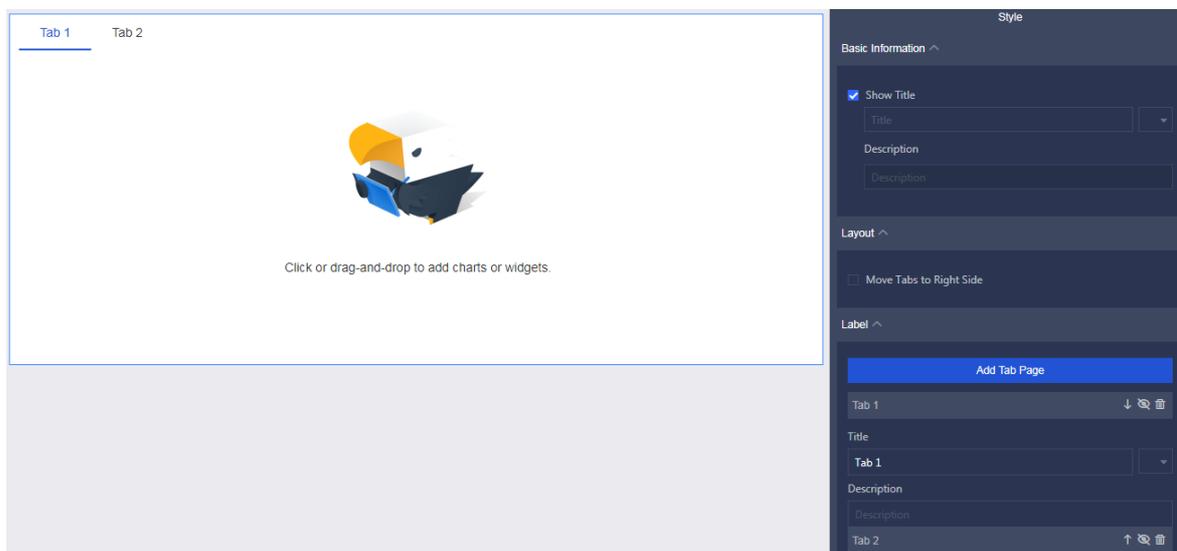
The tab widget enables you to display charts as tabs on a dashboard.

Prerequisites

The Quick BI service is purchased.

Procedure

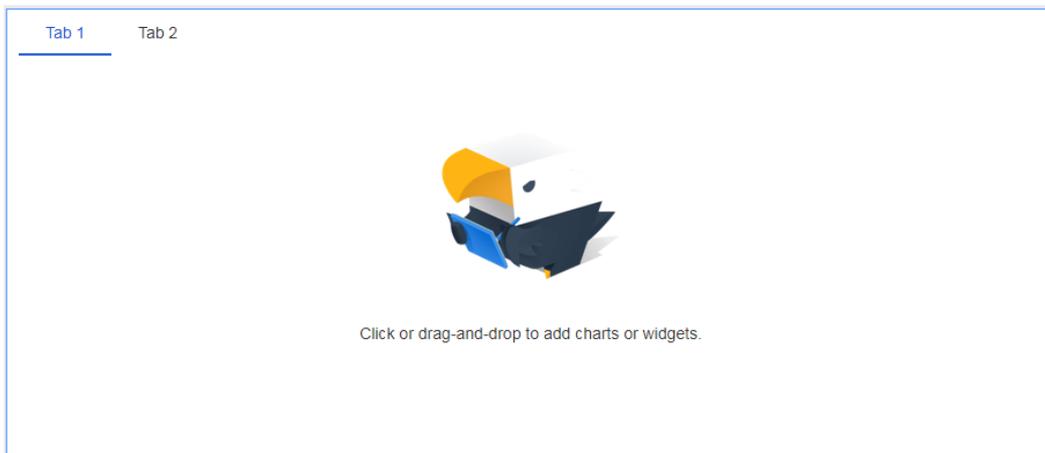
1. Go to the target dashboard.
2. Click the Tab icon.
3. Click the Style tab. Click **Add Tab Page** in the Label field.



4. Select a tab where you want to add charts, as shown in **Tabs**.

Click **Tab 1**. The tab name **Tab 1** becomes blue.

Tabs

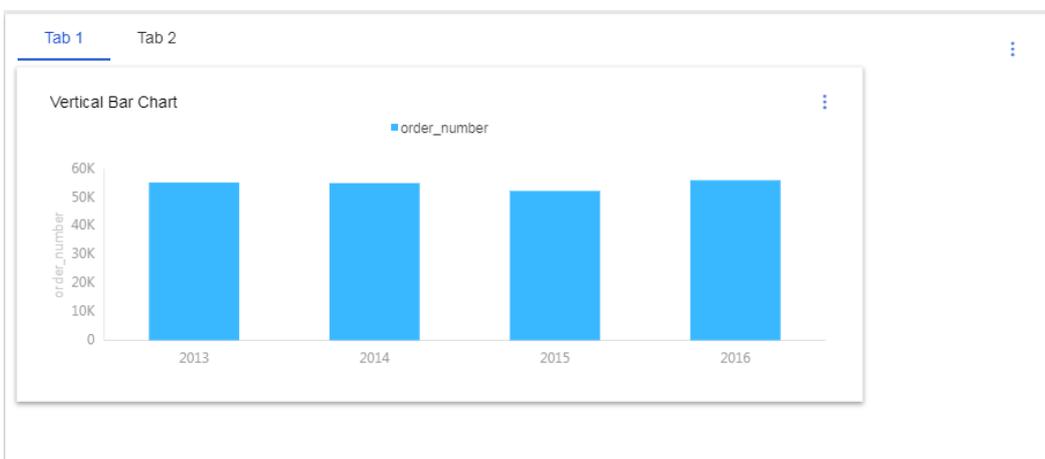


5. Click a chart icon to add a chart to Tab 1.



Add and configure a chart. **Sample tab** shows the tabs.

Sample tab



If you want to delete the current tab widget, click the **More** icon in the upper-right corner of the tab, and click **Delete**.

7.5.3.4.8.7. Image

The Image widget allows you to insert images into a dashboard. You can adjust the image position and display effects as needed.

Prerequisites

The Quick BI service is purchased.

Procedure

1. Go to the target dashboard.
2. Click the **Image** icon.
3. In the Style field, enter the URL and hyperlink of the image, and specify **Image Display**.



7.5.4. Create a chart on the dashboard

7.5.4.1. Create a line chart

A line chart shows data change trends and the interactions between multiple groups of data over a period of time. For example, you can use a line chart to analyze the sales volumes of a group of products or multiple groups of products over a period of time to forecast future sales volumes.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a line chart, you must specify **Value Axis (Mea.)**, **Category Axis (Dim.)**, and **Color Legend (Dim.)**.

- The category axis is determined by dimensions. You must specify at least one dimension, such as date, province, or product type.
- The value axis is determined by measures. You must specify at least one measure, such as order quantity.
- The color legend is determined by a dimension. You can select only one dimension, such as transportation method.

Note The color legend can be enabled when the value axis has only one measure.

The following example demonstrates the order quantity of each type of product in different provinces every year. A line chart and the `company_sales_record` dataset are used in this example.

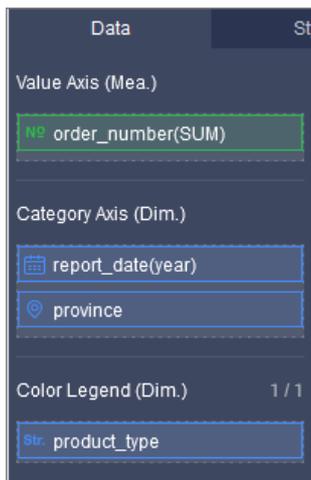
Procedure

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Line Chart** icon.
6. On the **Data** tab, select the required dimensions and measures.

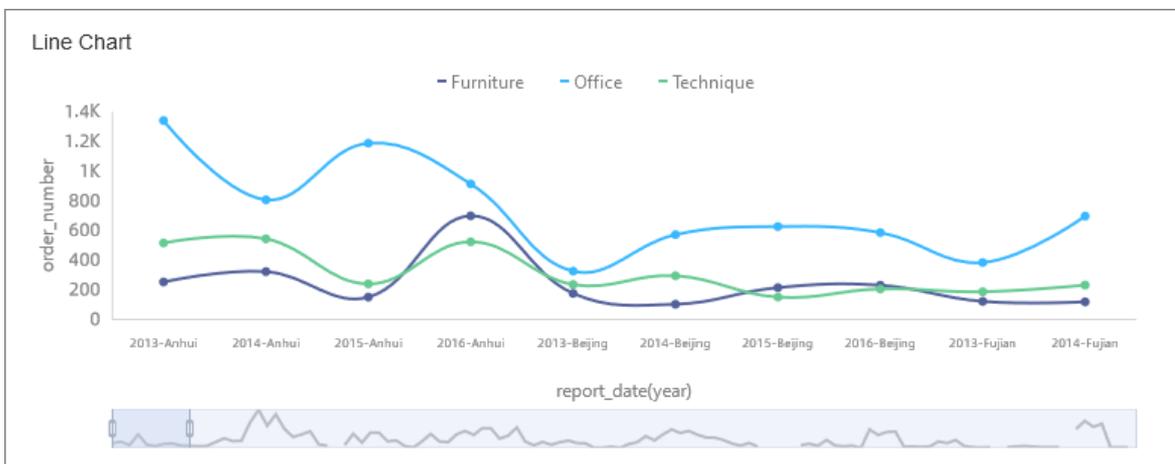
In the Dimensions list, find `order_date(year)` and `province` and add them to the Category Axis (Dim.) field. In the dimensions list, find `product_type` and add it to the Color Legend (Dim.) field. In the Measures list, find `order_number` and add it to the Value Axis (Mea.) field, as shown in [Specify fields for the line chart](#).

For information about how to convert the data type of a dimension, see [Edit a dimension](#).

Specify fields for the line chart



7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure the title, chart style, and maximum and minimum values of the line chart, as well as whether to show the axis, scroll bar, and scale.



- Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard and click **OK**.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.2. Create an area chart

An area chart shows data change trends at equal intervals with different sizes of area. In addition, you can use an area chart to analyze the interactions between multiple groups of data over a period of time. For example, analyze the sales volumes of a group of products or multiple groups of products over a period of time to forecast future sales volumes.

Background information

When you create an area chart, you must specify **Value Axis (Mea.)**, **Category Axis (Dim.)**, and **Color Legend (Dim.)**:

- The category axis is determined by dimensions. You must specify at least one dimension, such as date, province, or product type.
- The value axis is determined by measures. You must specify at least one measure, such as order_number.
- The color legend is determined by a dimension. You can select only one dimension, such as transportation method.

 **Note** The color legend is applicable only when the value axis has one measure.

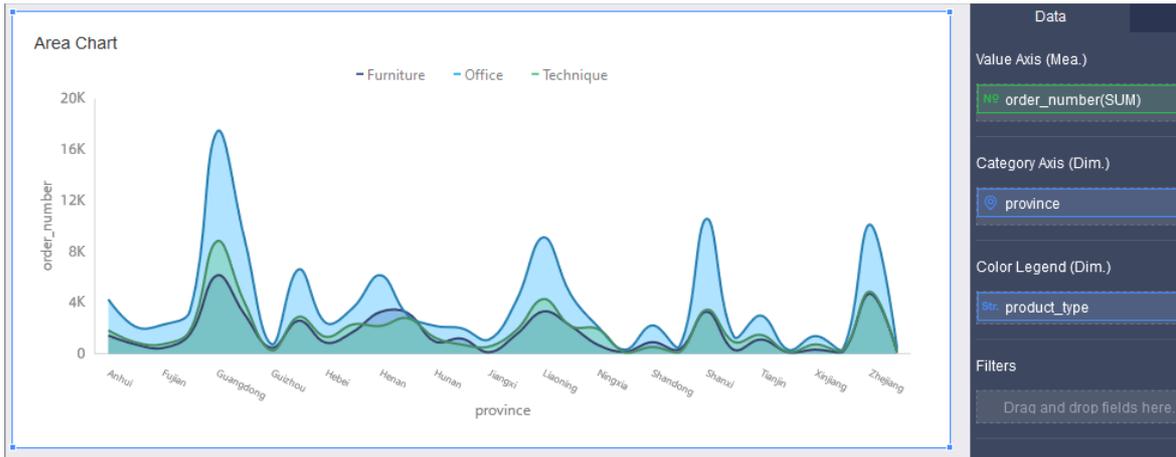
Procedure

The following example describes how to use an area chart to show the order quantity of each type of products in different provinces. The company_sales_record dataset is used in this example.

- [Log on to the Quick BI console.](#)
- In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
- On the **Datasets** page, click **MyDatasets** and find the *company_sales_record* dataset, then click the  icon in the Actions column.
- In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
- On the dashboard edit page, click **Area Chart**.
- On the **Data** tab, select required dimensions and measures.

In the Dimensions list, find **province** and drag it to the Category Axis (Dim.) field. In the Measures list, find **order_number** and drag it to the Value Axis (Mea.) field. Drag and drop **product_type** to the Color Legend (Dim.) field.

- Click **Update**.
- On the **Style** tab, you can change the title, chart type, show legend, and axis style of the chart. The following figure shows an updated area chart.



Note You can switch to another area chart type, such as stacked area chart or 100% stacked area chart as required.

9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard and click **OK**.

If you want to delete the chart, click **More Actions** in the upper-right corner of the chart and select **Delete**.

7.5.4.3. Create a vertical bar chart

A vertical bar chart demonstrates data changes over a period of time or comparisons between objects. For example, you can use a vertical bar chart to show the traffic flow in different periods of time at a crossing.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

Similar to a **Line chart**, a vertical bar chart consists of a category axis, a value axis, and a color legend.

This topic demonstrates how to use the filter and the dual y-axis function in a vertical bar chart.

- Scenario 1: Compare the shipping costs of different products in different provinces in East China.
- Scenario 2: Compare the order quantities and average profits of different products in different provinces.

You must specify at least one dimension, such as **province** or **product_type**, for the category axis. You must specify at least one measure, such as **order_number** or **profit_amt**, for the value axis. You can specify only one dimension for the color legend.

Note The color legend is applicable only when the value axis has one measure.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.

5. On the dashboard edit page, click the **Vertical Bar Chart** icon. Scenario 1: Compare the shipping costs of different products in different provinces in East China.

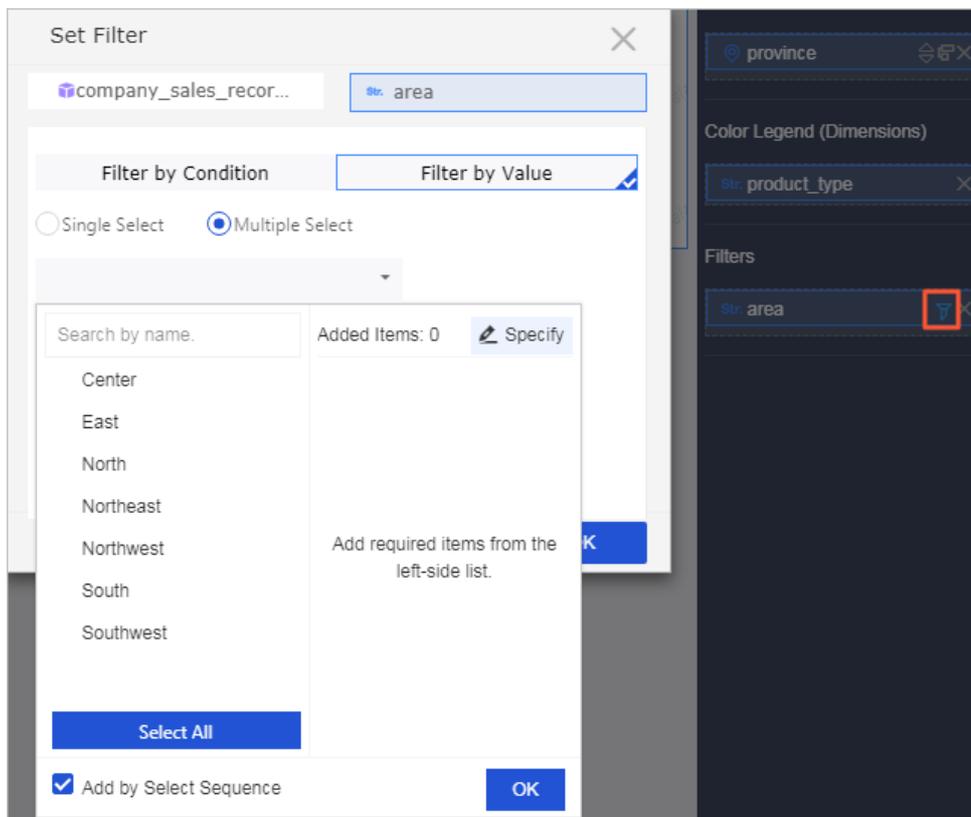
i. In the Dimensions list, find **area** and add it to the Filters field, as shown in **The filter**. Filter data of **East China** by using the Filter feature.

The filter



ii. Click the **Filter** icon.

iii. In the **Set Filter** dialog box, click **Filter by Value** and select **Multiple Select**.

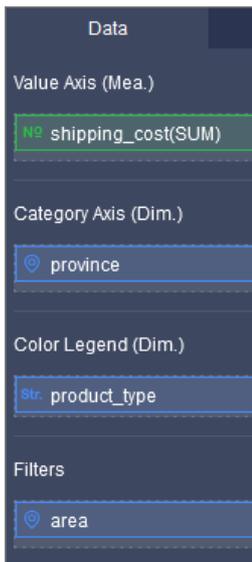


iv. Select **East** and click **OK**.

v. In the Dimensions list, find **province** and add it to the Category Axis (Dim.) field.

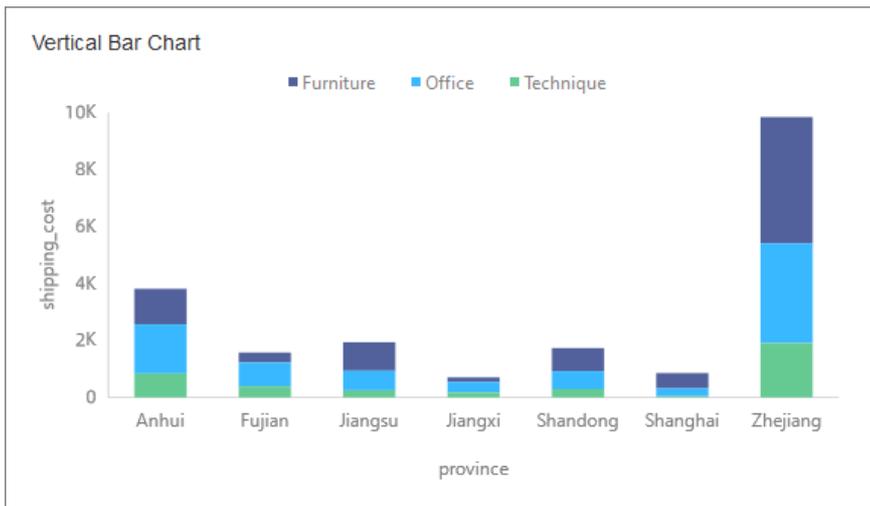
vi. In the Measures list, find **shipping_cost** and add it to the Value Axis (Mea.) field.

vii. Find **product_type** and add it to the Color Legend (Dim.) field.



viii. Click **Update**. The chart is updated.

ix. In the **Basic Information** section of the **Style** tab, select **Stacked Vertical Bar Chart** for **Chart Type**.



Scenario 2: Compare the order quantities and average profits of different products in different provinces. The `company_sales_record` dataset is used in the example.

Note

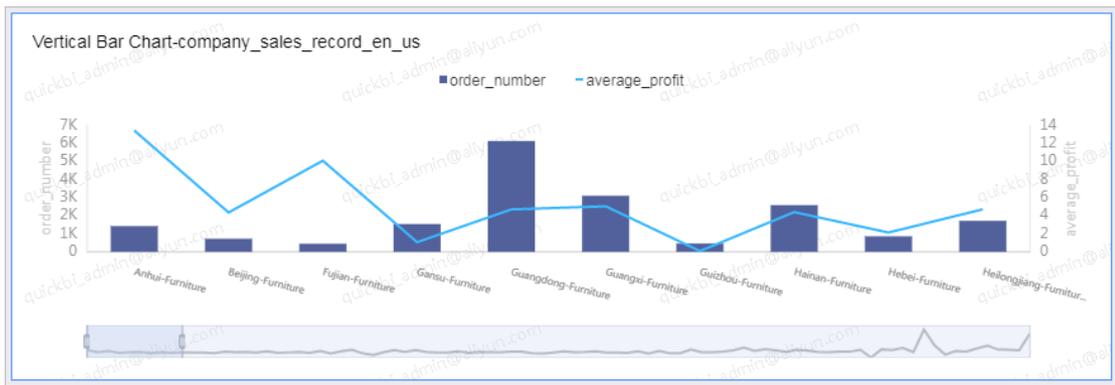
Data modeling may be required in this scenario. For information about data modeling, see [Add a calculated field](#).

i. On the **Data** tab, select required dimensions and measures.

In the Dimensions list, find **province** and **product_type** and add them to the Category Axis (Dim.) field. In the Measures list, find **order_number** and **average_profit** and add them to the Value Axis (Mea.) field.

ii. Click **Update**. The chart is updated.

iii. In the **Chart Type** section of the **Style** tab, select **Dual Y-Axis**.



iv. Click **Save** in the upper-right corner. In the **Save Dashboard** dialog box, enter a name for the dashboard and click **OK**.

7.5.4.4. Create a waterfall chart

A waterfall chart reflects data changes at different time periods or under the impact of different factors. It uses a combination of absolute and relative values and is suitable for business analysis and financial analysis.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

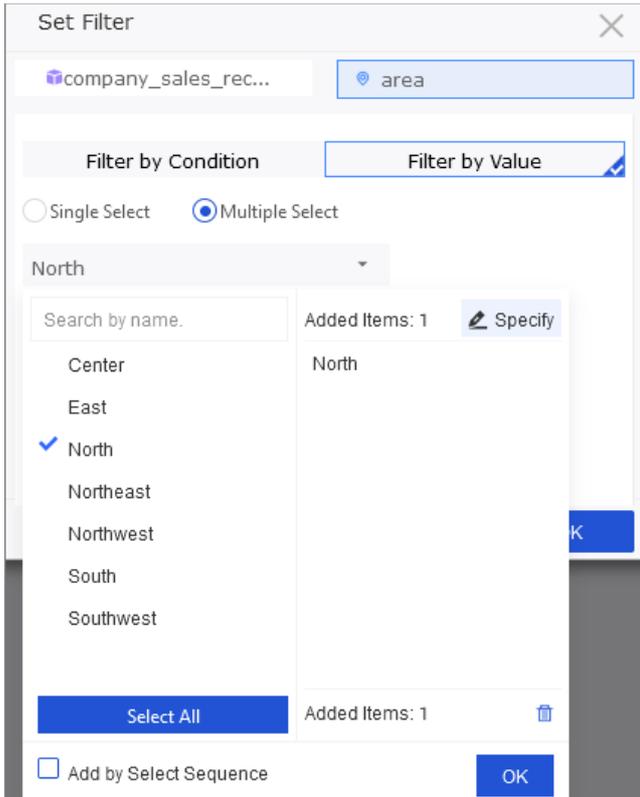
When you create a waterfall chart, you must specify **Category Axis (Dim.)** and **Value Axis (Mea.)**.

- The category axis is determined by dimensions. You must specify at least one dimension, such as province and product type.
- The value axis is determined by a measure. You can specify only one measure, such as order quantity or profit.

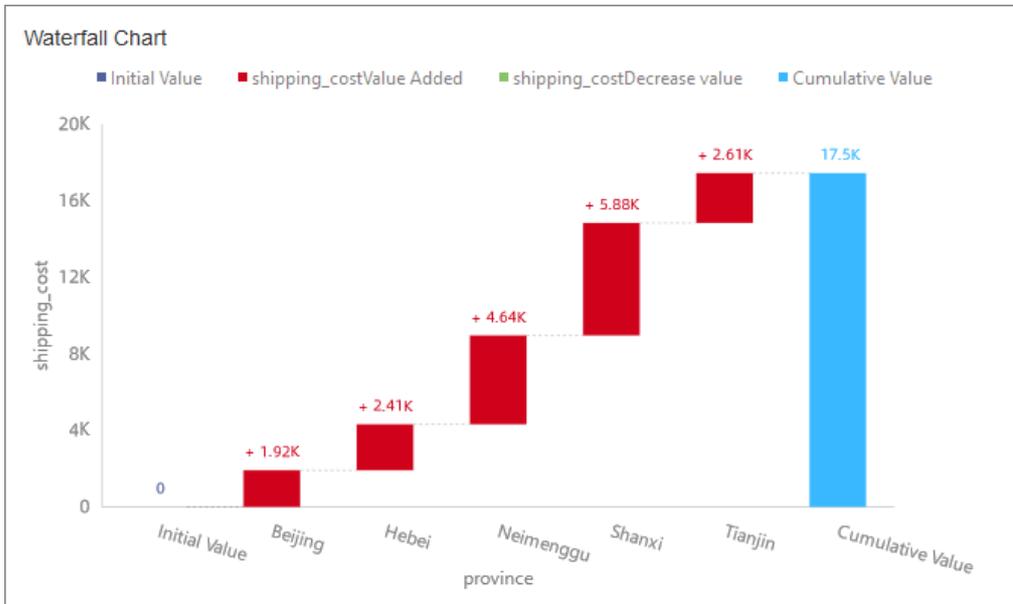
Create a waterfall chart

The following example describes how to use a waterfall chart to compare the shipping costs in the provinces of East China. The *company_sales_record* dataset is used in the example.

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, click **MyDatasets** and find the *company_sales_record* dataset, then click the  icon in the **Actions** column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Waterfall Chart** icon.
6. On the **Data** tab, select the required measures. In the **Dimensions** list, find **province** and drag it to the **Category Axis (Dim.)** field. In the **Measures** list, find **shipping_cost** and drag it to the **Value Axis (Mea.)** field. In the **Dimensions** list, drag and drop **area** to the **Filters** field.
7. Click the  icon next to **area**.



8. In the **Set Filter** dialog box, click **Filter by Value**, select **Multiple Select**, select **China (Beijing)** from the drop-down list, and then click **OK**.
9. Click **Update**. The chart is updated.
10. On the **Style** tab, configure parameters in the **Basic Information**, **Parameter Settings**, **Chart Type**, **Axes**, and **Series Settings** sections.



11. Click **Save** to save the dashboard. If you want to delete the chart, click the **More Actions** icon (⋮) in the upper-right corner of the chart and select **Delete**.

7.5.4.5. Create a horizontal bar chart

Similar to a vertical bar chart, a horizontal bar chart displays the differences between data of different categories.

Background information

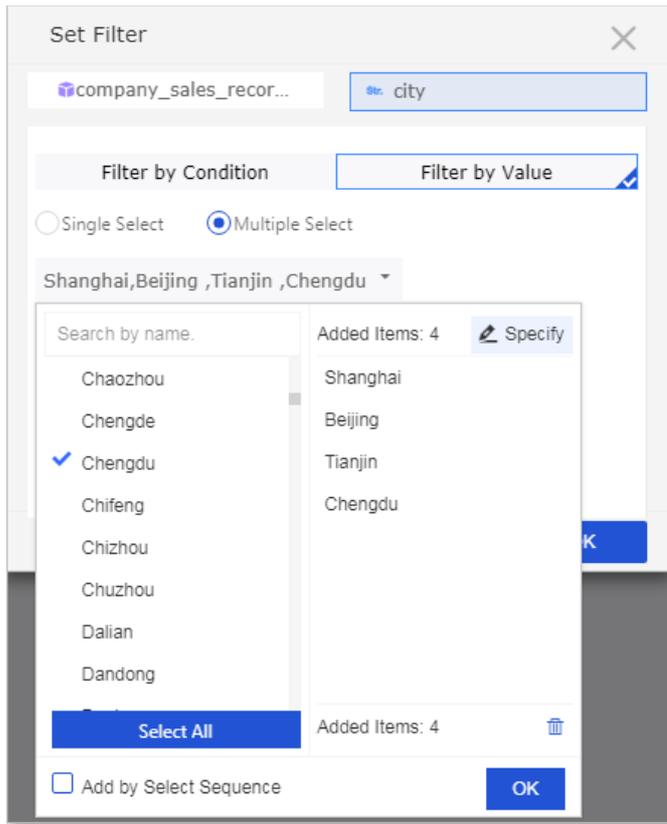
- A horizontal bar chart shows data changes in a period of time or compares differences between multiple groups of data. For example, you can use a horizontal bar chart to show the work progress of different members in a project group.
- When you create a horizontal bar chart, you must specify **Value Axis (Mea.)**, **Category Axis (Dim.)**, and **Color Legend (Dim.)**:
 - The value axis is determined by measures. You must specify at least one measure, such as order quantity or order price.
 - The category axis is determined by dimensions. You must specify at least one dimension, such as province or product type.
 - The color legend is determined by a dimension. You can select only one dimension, such as transportation method.

 **Note** The color legend is applicable only when the value axis has one measure.

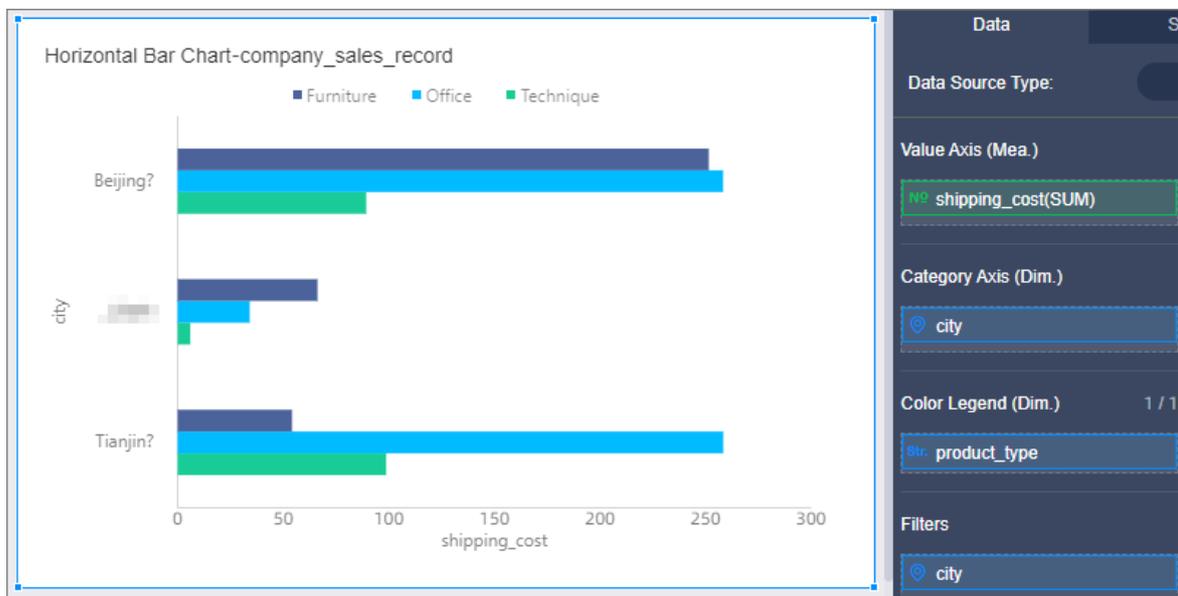
Procedure

The following example describes how to use a horizontal bar chart to compare the shipping costs of different products in different municipalities. The `company_sales_record` dataset is used in this example.

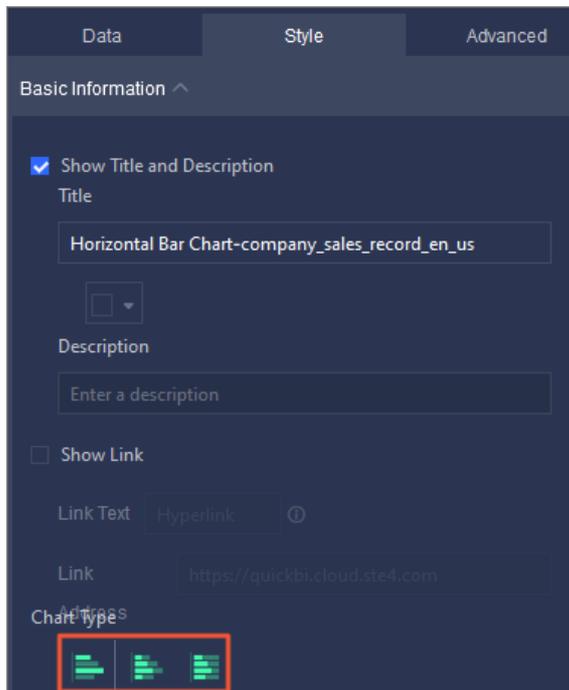
1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Horizontal Bar Chart** icon.
6. In the Dimensions list, find and add `city` to the Filters field and select four municipalities, as shown in the following figure.



7. In the dimensions list, find the **city** dimension and add it to the Category Axis (Dim.) field. In the measures list, find the **shipping_cost** measure and add it to the Value Axis (Mea.) field. In the dimensions list, find the **product_type** measure and add it to the Color Legend (Dim.) field.
8. Click **Update**. The chart is updated.



You can also switch the current chart to another bar chart type, such as stacked horizontal bar chart or 100% stacked horizontal bar chart.



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard and click **OK**.

If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.6. Create a progress bar

Similar to a gauge, a progress bar displays the progress of a specific metric.

Background information

A progress bar displays the progress of a task.

Progress pointers are determined by measures. You can select one to five measures, such as order quantity.

Before you use a progress bar, you must specify target values in the **Functionality** section on the **Style** tab.

Procedure

The following example describes how to use a progress bar to show order completion. The `company_sales_record` dataset is used in this example.

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Progress Bar** icon.
6. Click the **Data** tab and select the required measures.

In the Measures list, find `order_number` and add it to the Pointer (Mea.) field.

7. Click **Update**. The chart is updated.



8. Click the **Style** tab and configure a target value.

9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard and click OK.

If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.7. Create a combination chart

A combination chart compares data across multiple categories and magnitudes.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

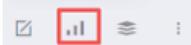
Context

A combination chart provides a dual y-axis to display data in multiple chart types, such as line chart, vertical bar chart, and area chart, in stacked and 100% stacked modes. For example, you can use a combination chart to display the change trends of different projects.

A combination chart consists of a category axis, a primary value axis, and a secondary value axis.

Notice You can specify at least one dimension for the category axis, such as the order_date(year). You can specify at least one measure each for the primary and secondary value axes, such as order price and profit. You can specify only one dimension for the color legend. The legend is applicable only when the primary or secondary value axis has one measure.

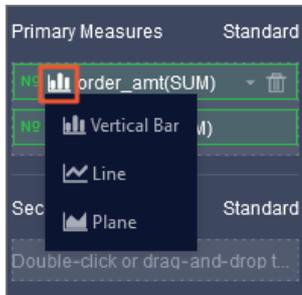
Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the Datasets page, find the *company_sales_record* dataset, click  in the Actions column. In the dialog box that appears, select **Standard** and click **OK**.
4. On the dashboard edit page, click the  icon.

- Click the **Data** tab and select the required dimensions and measures.

In the Dimensions list, find **order_date (year)** and add it to the Category Axis (Dim.) field. In the Measures list, find **order_amt** and add it to the Primary Measures field and **profit_amt** to the Secondary Measures field.

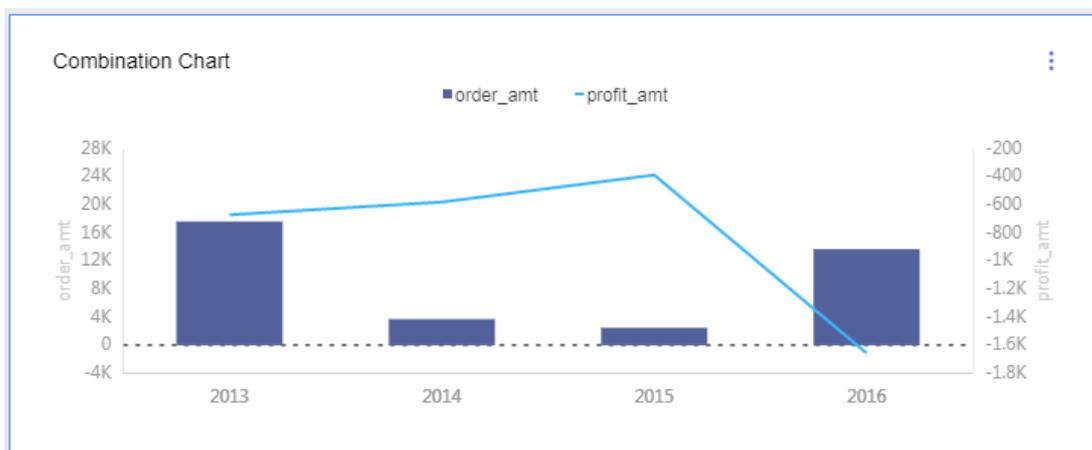
- In the Primary Measures and Secondary Measures fields, you can click the  icon to switch the chart type to vertical bar chart, line chart, or area chart.



- You can click **Standard** and select the required stack mode from the drop-down list.



- Click **Update**. The chart is updated.



- Click **Save**. The **Save Dashboard** dialog box appears. Specify Name and Save To.
- Click **OK** to save the dashboard.

If you want to delete the chart, perform the following steps:

- Click the  icon in the upper-right corner of the chart.
- Select **Delete**.

7.5.4.8. Create a pie chart

A pie chart shows data of different objects. Each object has a unique color or pattern. A pie chart shows the ratios of multiple values to the total amount. For example, you can use a pie chart to show the ratio of the income tax to the total personal income or the ratio of the sales volume of a car brand to the total sales volume.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a pie chart, you must specify **Labels (Dim.)** and **Central Angle (Mea.)**.

- Labels are determined by a dimension. You can specify only one dimension, such as area or product type.
- Central angles are determined by a measure. You can specify only one measure, such as order quantity or order price.

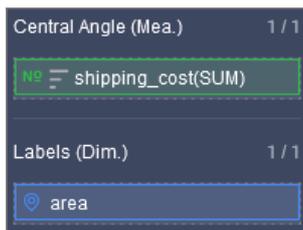
Compare the shipping costs in different regions. The `company_sales_record` dataset is used in the example.

Procedure

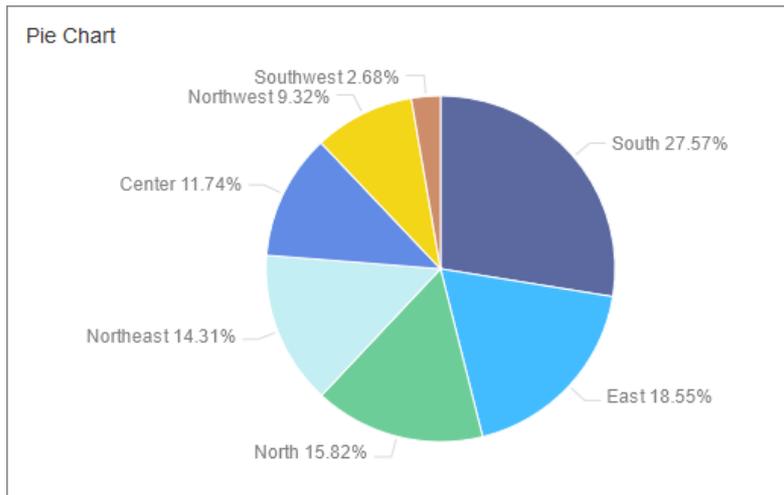
1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Pie Chart** icon.
6. Click the **Data** tab and select required dimensions and measures.

In the Dimensions list, find `area` and add it to the Labels (Dim.) field. In the Measures list, find `shipping_cost` and add it to the Central Angle (Mea.) field, as shown in [Specify fields for the pie chart.](#)

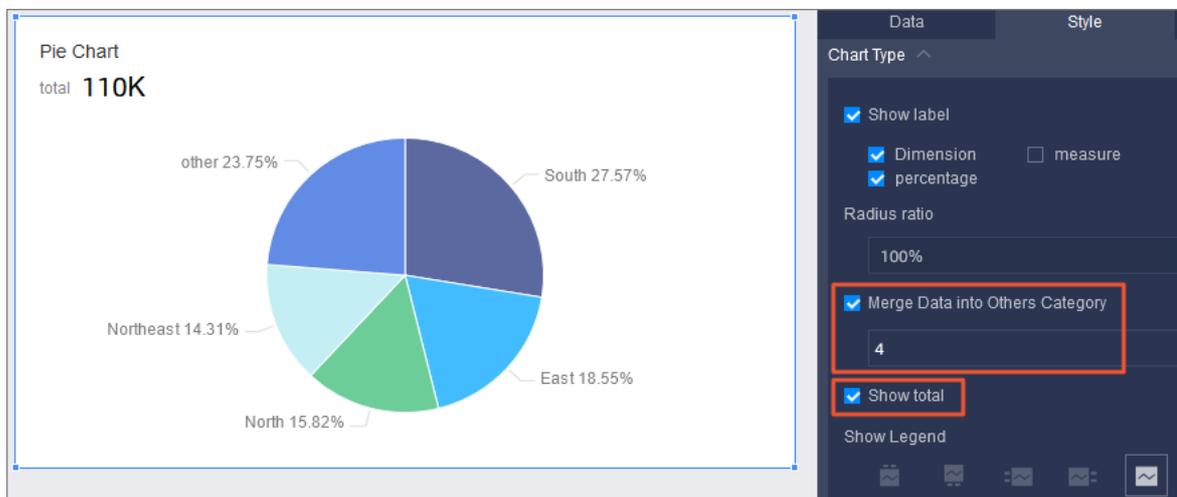
Specify fields for the pie chart



7. Click **Update**. The chart is updated.



- On the **Style** tab, you can change the title, labels, color, and value formats. In this example, select **Show total**, **Merge Data into Others Category**, and set the number to 4.



- Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
- Click **OK** to save the dashboard.

If you want to delete the chart, click the icon in the upper-right corner of the chart and select **Delete**.

7.5.4.9. Create a bubble map

A bubble map uses a map profile as its background and shows data distribution with bubbles of different sizes. It displays data metrics and their distribution in a country or region. For example, you can use a bubble map to display the volume of tourist arrivals at different destinations, or the average income in different regions.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A bubble map consists of geographic locations and bubbles in different sizes. Geographic locations are determined by a dimension, such as province. The bubble size is determined by one or more measures, such as shipping cost and order quantity.

When you create a bubble map, you must specify **Geo Location (Dim.)**, **Bubble Size (Mea.)**, and **Bubble color / metric**.

- The geo location is determined by the data dimension. You can select only one dimension and the dimension type must be geo, such as province.
- The Bubble color is determined by measures. You can select only one measure, such as the order_amt.
- The bubble size is determined by measures. You can specify one to five dimensions, such as shipping_cost and order_number.

Note Geo Bubble Map is unpublished. Bubble Map is provided.

The following example describes how to use a bubble map to compare the order quantities and average profits in different provinces. The company_sales_record dataset is used in this example.

Data modeling may be required for this scenario.

For information about data modeling, see [Add a calculated field](#).

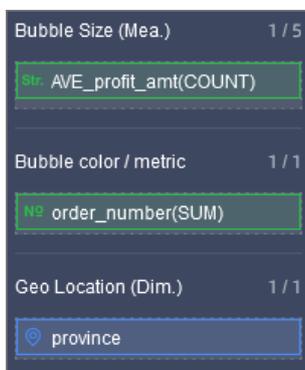
Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Bubble Map** icon.
6. On the **Data** tab, select required dimension and measures.

In the Dimensions list, find province and drag it to the Geo Location (Dim.) field. In the Measures list, find order_number and average_profit and drag them to the Bubble Size (Mea.) field. The following figure [Specify fields for the bubble map](#) is an example.

Note Make sure that you have converted the data type of the province and area dimensions from String to Geo. For information about how to change the data type of a dimension, see [Edit a dimension](#).

Specify fields for the bubble map



7. Click **Update**. The chart is updated.

8. On the **Style** tab, you can change the title, chart style, functionality, and data display format.



Click the **+** or **-** icon to zoom in or zoom out the map.

9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.

10. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.10. Create a colored map

Similar to a bubble map, a colored map displays data in a single color at different saturation levels. This visualization is useful to show the distribution of data across areas on a map.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a colored map, you must specify **Geo Location (Dim.)** and **Colorscale (Mea.)**.

- The geo location is determined by a dimension. You can select only one dimension and the dimension type must be geo, such as province.
- The colorscale is determined by measures. You can specify one to five dimensions, such as order price and profit.

Note Geo colored map is deprecated. The colored map is provided.

The following example describes how to use a colored map to compare the shipping costs, order prices, and profits in different regions. The `company_sales_record` dataset is used in this example.

Procedure

1. [Log on to the Quick BI console.](#)

2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, click **MyDatasets** and find the *company_sales_record* dataset, then click the  icon in the **Actions** column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Colored Map** icon.
6. On the **Data** tab, select required dimensions and measures. In the **Dimensions** list, find *area* and add it to the **Geo Location (Dim.)** field. In the **Measures** list, find *order_amt*, *profit_amt*, and *shipping_cost* and add them to the **Colorscale (Mea.)** field, as shown in [Specify fields for the colored map](#).

 **Note** Make sure that you have converted the data type of the dimension from **String** to **Geo**. For information about how to convert the data type of a dimension, see [Edit a dimension](#).

Specify fields for the colored map



7. Click **Update**. The chart is updated.
In a chart, you can adjust the map position and size. For more information, see [Create a bubble map](#).
8. Click **Save** in the upper-right corner. In the **Save Dashboard** dialog box, specify a name for the dashboard.
9. Click **OK** to save the dashboard.
If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.11. Create a cross table

A cross table displays the summary of a field and classifies field values into different groups. Aggregate functions used in a cross table include **SUM**, **AVG**, **COUNT**, **MAX**, and **MIN**.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a cross table, you must specify **Rows** and **Columns**.

- Rows are determined by dimensions, such as province and product type.
- Columns are determined by measures, such as order quantity and order price.

 **Note** Dimensions and measures are unlimited for each cross table.

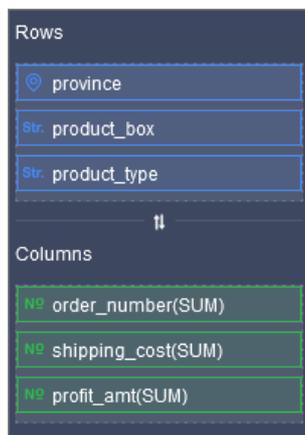
The following example describes how to use a cross table to compare the packaging, shipping costs, order quantities, and average profits of different products in different provinces. The `company_sales_record` dataset is used in this example.

Procedure

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Cross Table** icon.
6. On the **Data** tab, select the required dimensions and measures.

In the Dimensions list, find `province`, `product_type`, and `product_box` and add them to the Rows field. In the Measures list, find `order_number`, `shipping_cost`, and `profit_amt` and add them to the Columns field, as shown in [Specify fields for the cross table](#).

Specify fields for the cross table



7. Click **Update**. The chart is updated.
8. On the **Style** tab, perform the following steps:
 - o In the **Basic Information** section, configure Title and Show Link, as shown in the following figure.



- o In the **Display Settings** section, specify whether to custom table theme, show row numbers, merge cells that belong to the same category, freeze all columns by selecting Auto (Table Head), or freeze specific columns as required. After you configure the parameters, update the table. The following figure shows an updated chart.

The screenshot shows a data table with 8 rows and 7 columns. The columns are: province, product_box, product_type, order_number, shipping_cost, and profit_amt. The rows contain data for various product types like Furniture, Office, and Technique, with different box sizes (Huge Box, Huge Paperbag, Large Box). To the right of the table is a settings panel with tabs for Data, Style, and Advanced. The Data tab is active, showing options like 'Custom Table Theme', 'Show Row Numbers', 'Merge Same Cells', 'Freeze', 'Auto Table Head', 'Columns', 'Rows', 'Wrap Text', and 'Pagination'.

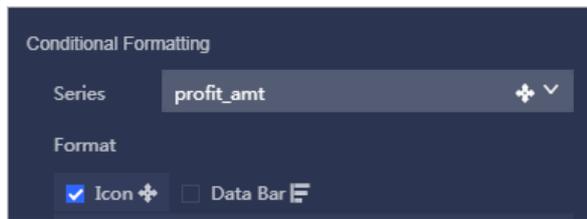
	province	product_box	product_type	order_number	shipping_cost	profit_amt
1			Furniture	207	274.32	2865.04
2		Huge Box	Office	21	47.02	716.56
3			Technique	134	166.5	-2267.499999999998
4		Huge Paperbag	Furniture	245	464.67000000000001	-4920.250000000001
5			Technique	46	30.06	252.77
6			Furniture	116	151.38	3580.77
7		Large Box	Office	128	94.95999999999998	2271.33
8			Technique	73	50.97	4714.17

- o In the **Functionality Settings** section, you can specify **Conditional Formatting**, **Sort Columns**, and **Show Totals**. Conditional formatting consists of **Icon** and **Data Bar**.

Icon:

- a. In the **Conditional Formatting** section, configure **Series** and **Format**.

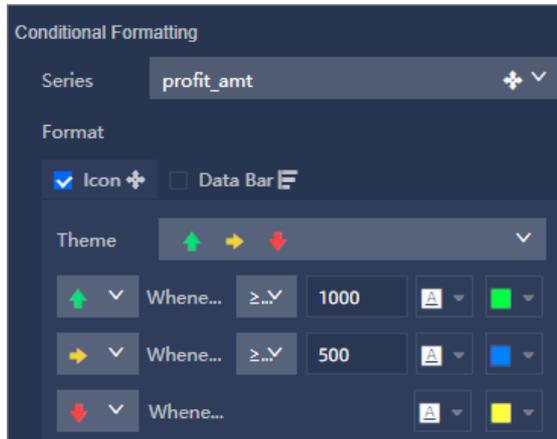
In this example, select profit_amt for **Series** and **Icon** for **Format**.



- b. Click the  icon and select an icon.



- c. Manually specify the values that you want to highlight, click the  icon, and then specify the font color and fill color for the values.



- If the data value is greater than 1000, the cell of the value is highlighted in green and a green upward arrow appears.
- If the data value is greater than or equal to 500 and less than or equal to 1000, the cell of the value is displayed in gray and an orange hyphen appears.
- If the data value is less than 500, the cell of the value is highlighted in yellow and a red downward arrow appears.

The updated chart is shown in the following figure.

	province	product_type	product_box	order_number	shipping_cost	profit_amt
1	Anhui	Furniture	Huge?Box	48	48.8	3799.59
2	Anhui	Furniture	Huge?Paperbag	45	89.3	-3033.57
3	Anhui	Furniture	Paperbag	96	8.76	956.73
4	Anhui	Office	Medium?Box	185	54.19999999999999	80.92999999999999
5	Anhui	Office	Paperbag	241	73.11999999999998	719.2300000000001
6	Anhui	Office	Small?Box	733	289.68999999999977	262.71999999999997
7	Anhui	Technique	Large?Box	27	26.47999999999999	6027.01
8	Anhui	Technique	Medium?Box	25	13.99	667.33
9	Anhui	Technique	Small?Box	244	56.410000000000004	2296.7499999999995
10	Anhui	Technique	Small?bag	11	2.96	-27.66
11	Beijing	Furniture	Huge?Box	4	30	-224.35
12	Beijing	Furniture	Huge?Paperbag	42	173.93	-1115.96

Data Bar

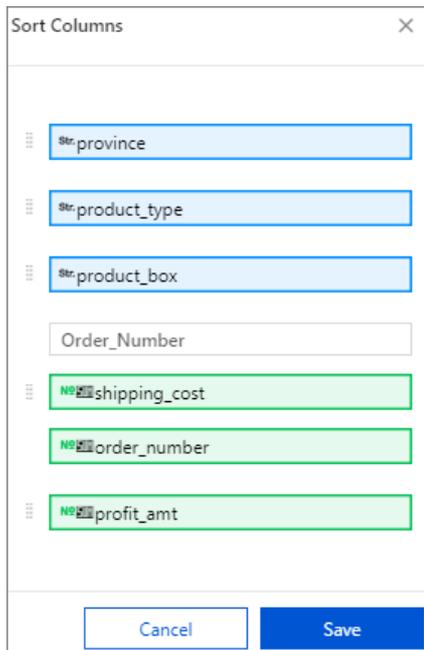
- a. Select a field and set the format to Data Bar.
- b. Configure the upper limit, lower limit, and color.

 Note Upper and lower limits can be configured automatically or manually.



Sort Columns

You can sort columns or group columns. When you group columns, specify the group names as shown in the following figure.



Show Totals

You can configure subtotals, totals, and total functions. The total functions can be aggregate functions such as SUM, AVG, and MAX. You can also customize a summation expression.

Note Before you configure subtotals, select **Merge Same Cells** in the **Display Settings** section.

- o In the **Series Settings** section, you can rename fields, specify the alignment mode and the number of records displayed on a single page.
9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
 10. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.12. Create a pivot table

A pivot table displays aggregates of variables and allows you to analyze data in a tree structure. One variable defines the values in the header row while the other variable defines the values in the header column. Aggregation methods include SUM, AVG, COUNT, MAX, and MIN.

Background information

Similar to **Cross tables**, a pivot table consists of **Rows (Dim.)** and **Values (Mea.)**:

- The rows are determined by dimensions, such as province and product_type.
- Values are determined by measures, such as order_number and profit_amt.

Notice Dimensions and measures are unlimited for each pivot table.

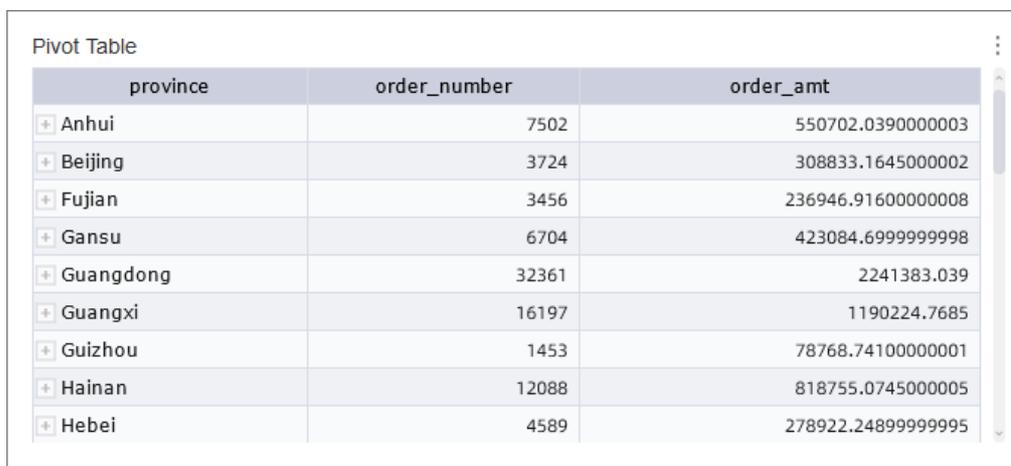
Procedure

The following example describes how to use a pivot table to show the packaging, order quantities, and order prices of different products in different provinces. The `company_sales_record` dataset is used in this example.

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Pivot Table** icon.
6. Click the **Data** tab and select the required dimensions and measures.

In the Dimensions list, find **province**, **product_type** and **product_box** and add them to the Rows (Dim.) field. In the Measures list, find **order_number** and **order_amt** and add them to the Values (Mea.) field.

7. Click **Update**. The chart is updated.



province	order_number	order_amt
Anhui	7502	550702.0390000003
Beijing	3724	308833.1645000002
Fujian	3456	236946.91600000008
Gansu	6704	423084.69999999998
Guangdong	32361	2241383.039
Guangxi	16197	1190224.7685
Guizhou	1453	78768.74100000001
Hainan	12088	818755.0745000005
Hebei	4589	278922.24899999995

8. Click **Save** in the upper-right corner. In the **Save Dashboard** dialog box, enter a name for the dashboard.
If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.13. Create a gauge

Similar to a dashboard in a car, a gauge shows the range of a specific metric. You can view the progress of a current task or determine whether a metric exceeds its value range in a gauge. For example, you can use a gauge to show the inventory status of a commodity, which helps you replenish the inventory in a timely manner.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

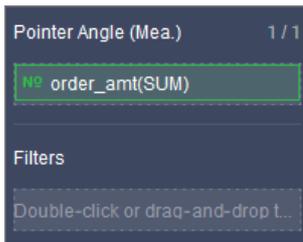
When you create a gauge, you must specify **Pointer Angle (Mea.)**. You can specify only one measure for this field, such as discount or profit.

The following example describes how to use a gauge to show order prices. The `company_sales_record` dataset is used in this example.

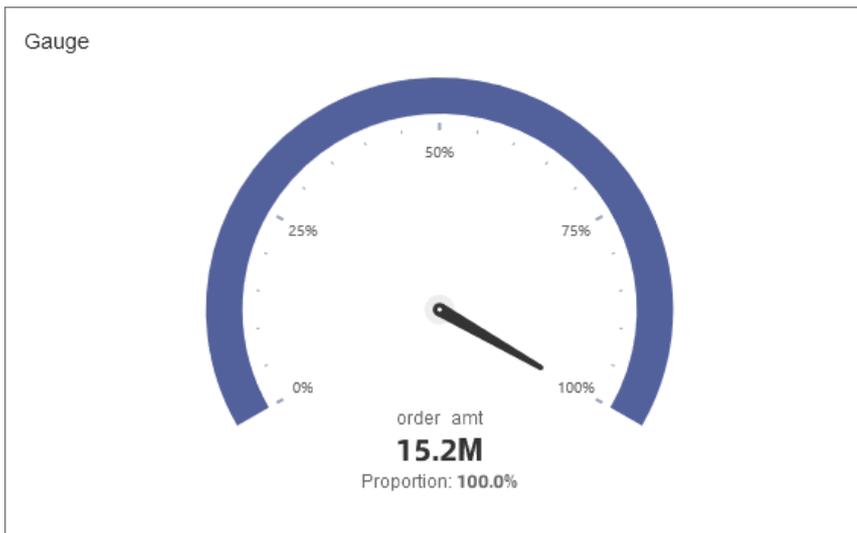
Procedure

1. On the Data Sources page, click Create Data Source in the upper-right corner. For more information, see [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Gauge** icon.
6. On the **Data** tab, select the required measure.

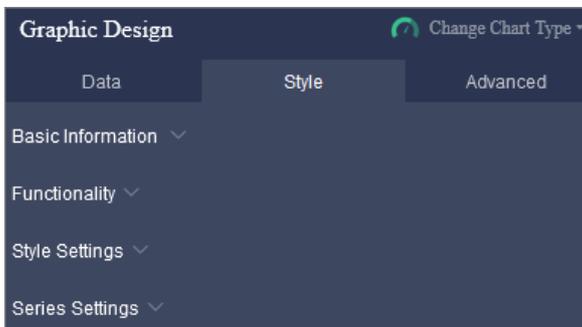
In the Measures list, find *order_amt* and add it to the Pointer Angle (Mea.) or Tooltip (Mea.) field.



7. Click **Update**. The chart is updated.



8. On the **Style** tab, configure the title, minimum and maximum values, legend position, and number format of the chart, as well as whether to display scales.



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.14. Create a radar chart

A radar chart shows numbers or ratios obtained from analysis. It allows you to learn about the changes and trends of data metrics. For example, you can use a radar chart to show the sales volumes in different regions.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A radar chart consists of labels and label lengths. Labels are determined by dimensions, such as product type. Label lengths are determined by measures, such as shipping cost.

You can specify one or two dimensions for labels. We recommend that the dimension values be greater than or equal to three and the number of dimensions be less than or equal to 12. You can specify at least one measure for label lengths.

The following example describes how to use a radar chart to compare the order quantities and order prices in different regions. The `company_sales_record` dataset is used in this example.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Radar Chart** icon.
6. On the **Data** tab, select required dimensions and measures.

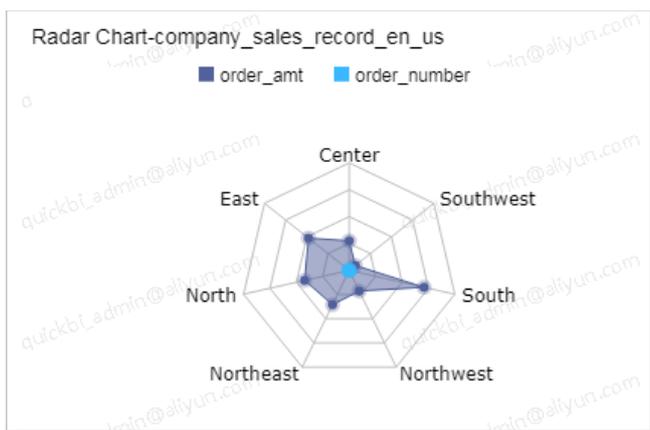
In the Dimensions list, find `area` and add it to the Labels (Dim.) field. In the Measures list, find `order_number` and `order_amt` and add them to the Length (Mea.) field, as shown in [Specify fields for the radar chart](#).

Specify fields for the radar chart



7. Click **Update**. The chart is updated.
8. On the **Style** tab, change the title, layout, and legend position of the radar chart, as shown in [Radar chart](#).

Radar chart



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.
If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.15. Create a facet scatter chart

A facet scatter chart shows data correlation and distribution.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a facet scatter chart, you must specify **Y-Axis (Mea.)**, **X-Axis (Mea.)**, and **Color Legend (Dim.)**.

- The y-axis is determined by a measure. You can specify only one measure.
- The x-axis is determined by measures. You can specify one to three measures.
- The color legend is determined by dimensions. You can specify only one dimension. The dimension can have a maximum of 1,000 values.

For example, if you add **product_type** to the **Color Legend (Dim.)** field, a maximum of 1,000 product types are allowed.

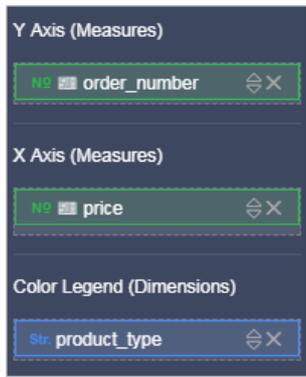
The following example describes how to use a facet scatter chart to compare the unit prices and order quantities of different products. The **company_sales_record** dataset is used in this example.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the  icon.
6. On the **Data** tab, select the required dimension and measures.

In the Dimensions list, find `product_type` and add it to the Color Legend (Dim.) field. In the Measures list, find `order_number` and add it to the Y-Axis (Mea.) field. Find `price` and add it to the X-Axis (Mea.) field, as shown in [Specify fields for the facet scatter chart](#).

Specify fields for the facet scatter chart



7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure the title, axis title, and legend position of the facet scatter chart, as shown in [Facet scatter chart](#).

Facet scatter chart



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the icon in the upper-right corner of the chart and select **Delete**.

7.5.4.16. Create a scatter chart

A scatter chart shows the correlation and distribution of data. It is generally used for data aggregation across categories.

Context

A scatter chart takes one variable as the x-axis and the other variable as the y-axis, which represents the general trend of the dependent variable (Y) that changes in respect to the independent variable (X). Scatter charts can also show data distribution from category and color.

A scatter chart consists of a y-axis, an x-axis, a category, and a color.

You can specify only one dimension or measure for the x-axis.

You can specify only one measure for the y-axis.

You can specify only one dimension or measure for the category.

The following example describes how to use a scatter chart to compare the prices and order quantities in different regions. The `company_sales_record` dataset is used in this example.

Procedure

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, click **MyDatasets** and find the `company_sales_record` dataset, then click the  icon in the **Actions** column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the  icon.
6. On the **Data** tab, select the required dimensions and measures.
 - In the **Dimensions** list, find the `area`, and drag it to the **Category (Dim.)** and **Color (Dim. or Mea.)** field.
 - In the **Measures** list, find `price`, and double-click or drag it to the **Y axis (Mea.)** field.
 - In the **Measures** list, find `order_number`, double-click or drag it to the **X axis (Dim. or Mea.)** field.



7. Click **Update**. The chart is updated.
8. On the **Style** tab, you can configure the **Title**, **Show Legend**, **Axes**, **Enable Four Quadrants** information and data display format. The **Scatter chart** is an example.

Scatter chart



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the  icon in the upper-right corner of the chart and select **Delete**.

7.5.4.17. Create a bubble chart

A bubble chart displays data distribution and aggregation by placing proportionally sized bubbles in corresponding locations.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Background information

A bubble chart consists of an x-axis, a y-axis, a category, bubble colors, and bubble sizes. The x-axis is determined by a dimension or measure. The y-axis is determined by a measure. The category is determined by a dimension. The color is determined by a dimension or measure. The size is determined by a measure.

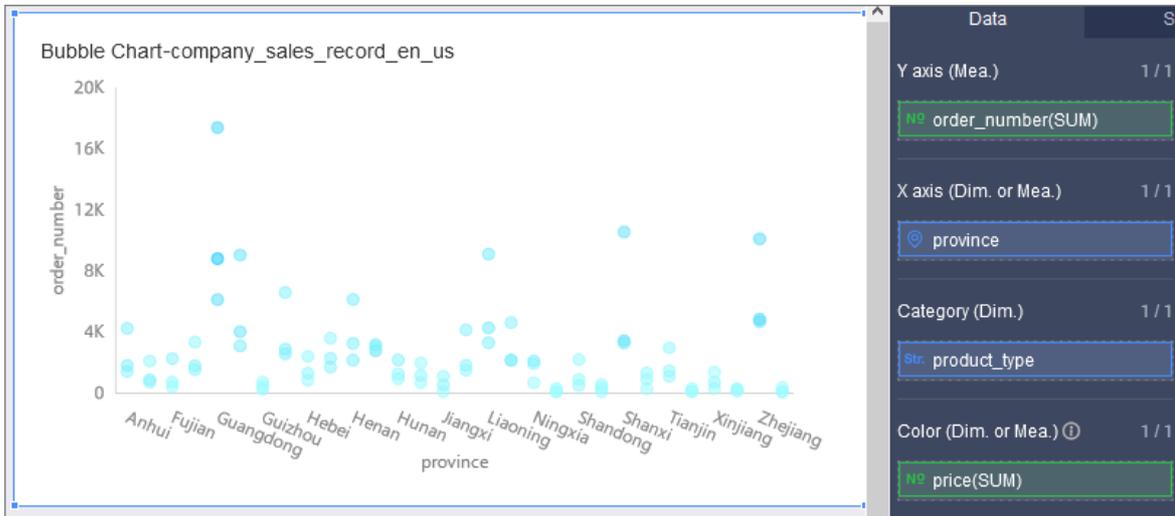
- You can specify only one dimension or measure for the x-axis.
- You can specify only one measure for the y-axis.
- You can specify only one dimension for the category.
- You can specify only one dimension or measure for the color.
- You can specify only one measure for the size.

Procedure

The following example describes how to use a bubble chart to compare the order quantities, prices and product types in different provinces. The `company_sales_record` dataset is used in this example.

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
In this example, **Standard** is selected.
5. On the dashboard edit page, click the  icon.

- On the **Data** tab, select the required dimension and measure.
 - In the dimensions list, double-click or drag the province and product_type fields to the **X axis (Dim. or Mea.)** or **Category (Dim.)** section.
 - In the measures list, double-click or drag the order_number and price fields to the **Y axis (Mea.)** and **Color (Dim. or Mea.)** section.
- Click **Update**. The chart is updated.



- On the **Style** tab, you can change the title, labels, legends, axes, functionality, and value formats.
- On the **Advanced** tab, configure the following parameters.
- Click **Save** in the upper-right corner of the page. In the Save Dashboard dialog box, specify the name and save path for the dashboard, and click OK.

If you want to delete the chart, click the icon in the upper-right corner of the chart and select **Delete**.

7.5.4.18. Create a funnel chart

A funnel chart is suitable for analyzing standard, long-running, and multi-flow business processes. By comparing business data from different stages, funnel charts allow you to detect and analyze problems. You can also use a funnel chart to show the conversion rates between the stages of a business process, such as the percentage of visitors that become paying customers for a shopping website. Funnel charts are ideal for business process analysis.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a funnel chart, you must specify **Tier Area (Mea.)** and **Funnel layering / dimension**.

- The number of funnel layers is determined by a dimension. You can specify only one dimension, such as area. You can also leave this field empty.
- The tier area is determined by measures. You can specify 1 to 10 measures, such as the order price.

Notice You can configure the funnel chart in the following two methods:

- Specify one dimension for Funnel layering / dimension and one measure for Tier Area (Mea.). This type of funnel charts is suitable for analyzing the business data of a specific stage. You can easily identify exceptions.
- Specify 1 to 10 measures for Tier Area (Mea.) and leave the Funnel layering / dimension field empty. This type of funnel chart is suitable for comparing business data of multiple stages.

The following example describes how to use a funnel chart to compare the order price in different regions. The `company_sales_record` dataset is used in this example.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Funnel Chart** icon.
6. On the **Data** tab, select the required dimensions and measures.

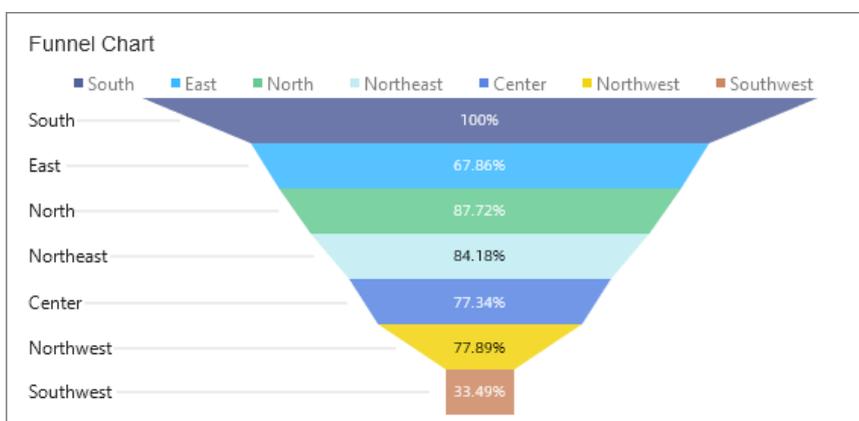
In the Dimensions list, find `area` and add it to the Funnel layering / dimension field. In the Measures list, find `order_amt` and add it to the Tier Area (Mea.) field, as shown in [Specify fields for the funnel chart.](#)

Specify fields for the funnel chart



7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure the title and legend position of the funnel chart, as shown in [Funnel chart.](#)

Funnel chart



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.19. Create a kanban

A kanban displays data such as sales or operation status, to help you formulate solutions based on the data. Kanbans help you discover and solve problems.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A kanban consists of metrics and labels. Labels are determined by a dimension, such as area. Metrics are determined by measures, such as order quantity and order price.

You can specify only one dimension for labels and one to ten measures for metrics.

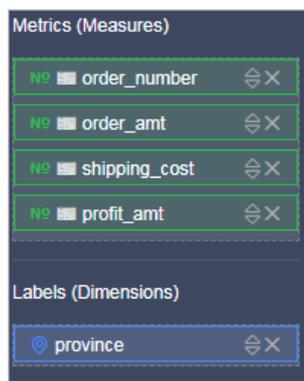
The following example describes how to use a kanban to compare the order quantities, order price, shipping costs, and profits in different provinces. The `company_sales_record` dataset is used in this example.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Kanban** icon.
6. On the **Data** tab, select required dimension and measures.

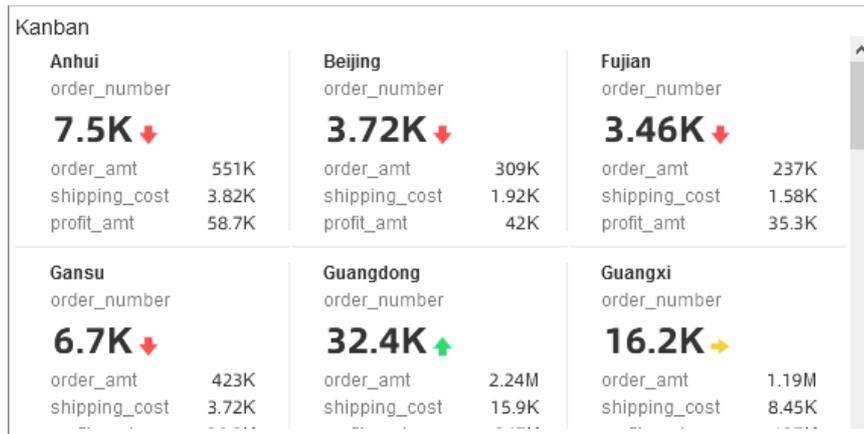
In the Dimensions list, find province and add it to the Labels (Dim.) field. In the Measures list, find order_number, order_amt, shipping_cost, and profit_amt and add them to the Metrics (Mea.) field, as shown in [Specify fields for the kanban](#).

Specify fields for the kanban



7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure the title, background, font size, the number of indicators displayed in each row, conditional formatting, and data display format of the kanban, as shown in [Kanban](#).

Kanban



- Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
- Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.20. Create an indicator trend chart

An indicator trend graph displays multiple indicators over a period of time. By default, the latest data record of each indicator is displayed.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

You can specify only one date dimension for Date (Dimensions). You must specify at least one measure for Indicator (Measures).

The following example describes how to use an indicator trend graph to compare the order quantities, order price, shipping costs, and profits in different provinces in a specified period of time. The `company_sales_record` dataset is used in this example.

Create an indicator trend chart

- [Log on to the Quick BI console.](#)
- In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
- On the **Datasets** page, click **MyDatasets**, find the `company_sales_record` dataset, and click the  icon in the Actions column.
- In the **Create Dashboard** dialog box, click **Standard** or **Full Screen**, and click **OK**.
- On the dashboard edit page, click the **Indicator trend graph** icon.
- In the Dimensions list, find `order_date(day)` and add it to the **Date (Dimensions)** field. In the Measures list, find and add `order_number`, `order_amt`, `shipping_cost`, and `profit_amt` to the **Indicator (Mea.)** field.



7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure parameters in the **Basic Information**, **Chart Settings**, **Indicator Settings**, **Functionality Settings**, and **Series Settings** sections.
9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete a chart, move the pointer over the chart, click the  icon in the upper-right corner, and select **Delete**.

Configure the parameters on the Style tab

1. In the **Basic Information** section, configure Title, Description, and Show Link.

 **Note**

- This example uses light color as the background color.
- If you want users to be redirected to a report or an external page, select Show Link and specify Link Text and Link Address.

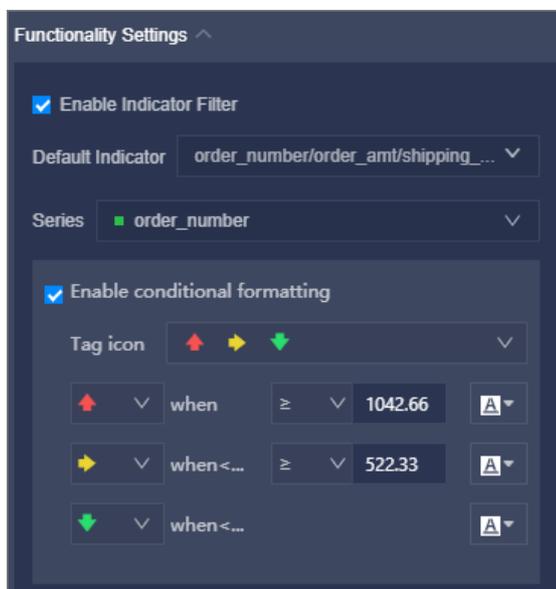
2. In the **Chart Settings** section, select Show Trend by Date, and specify a trend chart style, preview mode, and layout for multiple indicators.
 - If you select Single Choice for Select Preview Mode, the trend chart displays the trend of only one measure indicator. If you select Multiple Choice for Select Preview Mode, the trend chart displays the trend of multiple measure indicators.
 - If you select Displayed in One Line for Layout for Multiple Indicators and the indicators cannot be displayed in one line, you can slide the chart to view all indicators. If you select Displayed in Multiple Lines for Layout for Multiple Indicators, indicators are displayed in multiple lines.
3. In the **Indicator Settings** section, specify the number of indicators in each row, the font color for the primary indicator value, and the alignment mode of indicator blocks. In this example, Maximum Indicators Each Row is set to 4.
4. In the **Functionality Settings** section, configure Enable Indicator Filter and conditional formatting.

After you select Enable Indicator Filter, an indicator filter icon is displayed in the upper-right corner of the chart. You can click the icon to select the indicators that you want to display.

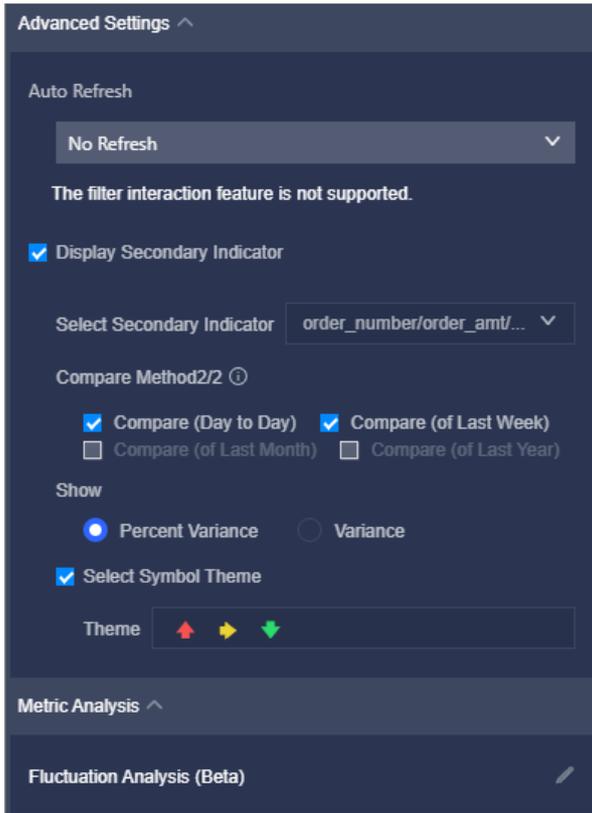


Perform the following steps to configure conditional formatting:

- i. Select the required measure from the Series drop-down list. Then, select **Enable Indicator Filter** and select icon themes from the drop-down list of the Tag icon.



- ii. Specify the rules for marking data, the icon style, and font color.
5. In the **Series Settings** section, set the measure alias, description, prefix and suffix of indicator values, and data display format.
 6. On the **Advanced** tab, select Display Secondary Indicator. Select the required measure indicator from the **Select Secondary Indicator** drop-down list, and set the comparison content, display content, and symbol theme.



The following figure shows an updated indicator trend graph.



7.5.4.21. Create a treemap

A treemap compares the proportions of metrics of an object.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A treemap displays labels in different sizes based on the measure. Labels are determined by a dimension, such as packaging. The size of each rectangle label is determined by a measure, such as shipping costs.

You can specify only one dimension for labels and one measure for sizes. The dimension must have a maximum of 12 values.

The following example uses the `company_sales_record` dataset to describe how to use a treemap to compare the order quantities of different products.

Procedure

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. In the Datasets page that appears, find the `company_sales_record` dataset, click the **Create Dashboard** icon in the Actions column. In the dialog box that appears, select **Standard** and click OK.
4. On the dashboard edit page, click the **Treemap** icon.
5. Click the **Data** tab and select the required dimension and measures.

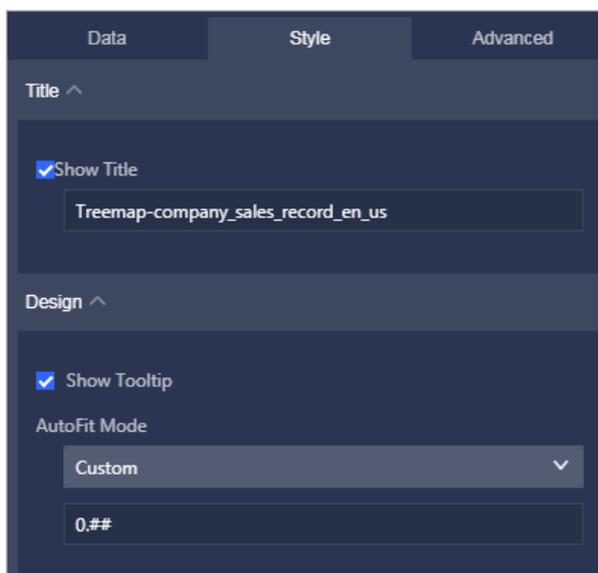
In the Dimensions list, find and add `product_type` to the Labels (Dim.) field. In the Measures list, find and add `order_number` to the Size (Mea.) field, as shown in [Specify fields for the treemap](#).

Specify fields for the treemap



6. Click **Update**. The chart is updated.
7. Click the **Style** tab and configure whether to show the title and tooltip of the treemap, as shown in [Treemap](#).

Treemap





8. Click **Save** in the upper-right corner. In the Save Dashboard dialog box that appears, enter a name for the dashboard.
9. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.22. Create a polar diagram

A polar diagram displays data changes over a period of time or compares metric values. It is ideal for comparing data of different objects, for example, data across different regions.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

Similar to a [pie chart](#), a polar diagram consists of multiple slices. Slice labels are determined by a dimension, such as area or product type. The arc radius of each slice is determined by a measure, such as order quantity or order price.

You can specify only one dimension for slice labels and only one measure for arc radii.

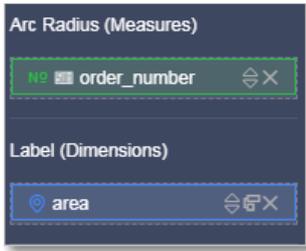
The following example uses the `company_sales_record` dataset to describe how to use a polar diagram to compare the order quantities in different regions. The number of regions must be greater than or equal to 3 and less than or equal to 12.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the Datasets page that appears, find the `company_sales_record` dataset, click the **Create Dashboard** icon in the Actions column. In the dialog box that appears, select **Standard** and click **OK**.
4. On the dashboard edit page, click the **Polar Diagram** icon.
5. Click the **Data** tab and select the required dimension and measure.

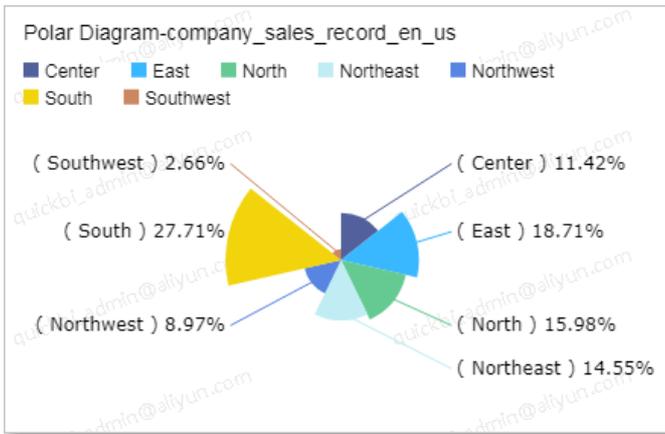
In the Dimensions list, find and add `area` to the Label (Dim.) field. In the Measures list, find and add `order_number` to the Arc Radius (Mea.) field, as shown in [Specify fields for the polar diagram](#).

Specify fields for the polar diagram

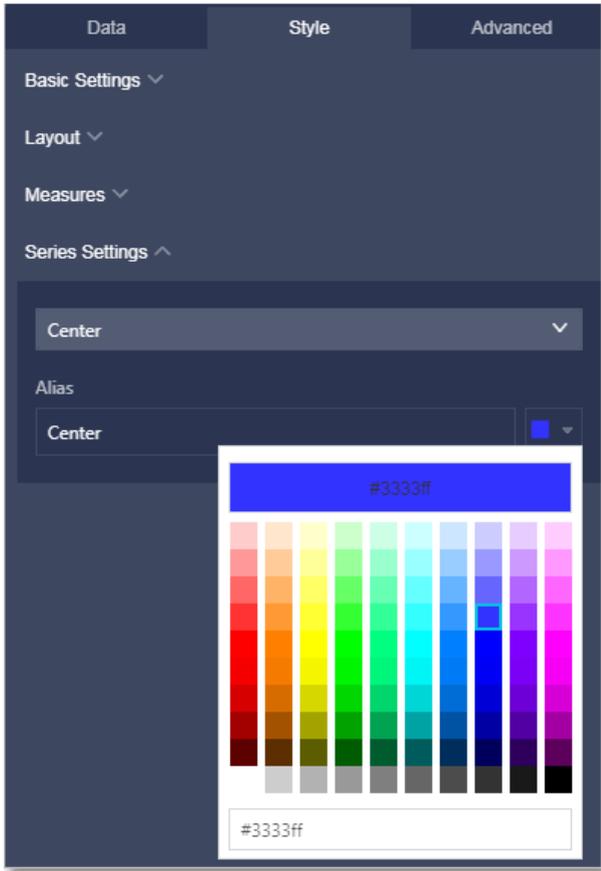


6. Click **Update**. The chart is updated.
7. Click the **Style** tab, change the title of the polar diagram, and specify the legend position, as shown in **Polar diagram**.

Polar diagram



8. Click the **Style** tab and change the legend colors in the **Series Settings** section.
Change legend colors



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box that appears, enter a name for the dashboard.
10. Click **OK** to save the dashboard.
If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.23. Create a word cloud

A word cloud displays the frequency of words that appear in a dataset. It is ideal for creating user personas and user tags.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

When you create a word cloud, you must specify **Word Size (Mea.)** and **Word (Dim.)**:

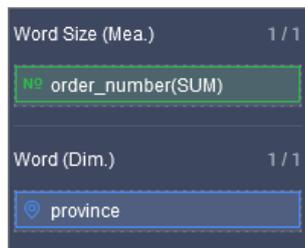
- The word size is determined by a measure. You can select only one measure, such as profit amount or unit price.
- The label is determined by a dimension. You can select only one dimension, such as customer name or product name.

The following example describes how to use a word cloud to compare the order quantities in different provinces. The company_sales_record dataset is used in this example.

Procedure

1. Log on to the Quick BI console.
2. In the left-side navigation pane of the **Workspace** tab, click the **Datasets** icon.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Word Cloud** icon.
6. On the **Data** tab, select the required dimension and measure. In the Dimensions list, find province and drag it to the Word (Dim.) field. In the Measures list, find order_number and drag it to the Word Size (Mea.) field.

Specify fields for the word cloud



7. Click **Update**. The chart is updated.
8. On the **Style** tab, change the title of the chart, word cloud shape and value display formats. The following figure **Word cloud** is an example.

Word cloud



9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More Actions** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.24. Create a butterfly chart

A butterfly chart is the combination of a comparison chart and a funnel chart. A comparison chart is used to compare metrics of two different objects, such as the income and education level differences between residents in two cities. A funnel chart shows the conversion rates between the stages of business processes, such as the percentage of visitors that become paying customers for a shopping website. Funnel charts are ideal for business process analysis.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

- A comparison chart compares different metrics between two objects,
-

A butterfly chart combines the features of a comparison chart and a funnel chart.

- If no conversion relationship exists, the butterfly chart functions the same as a comparison chart.
- If a conversion relationship exists between metrics and only one comparison subject is defined, the butterfly chart functions the same as a funnel chart.
- If a conversion relationship exists between metrics and multiple comparison subject are defined, the advantages of a butterfly chart is fully exploited.

If you want to compare the percentage of the migrant populations, employment rates, and commercial housing transactions in Beijing and Shanghai, the butterfly chart shows the values of the metrics for the two cities and also the conversion rates between the metrics.

When you create a butterfly chart, you must specify **Tier Area (Mea.)** and **Comparison Metrics (Dim.)**:

- The comparison metric is determined by a dimension. You can select only one dimension, such as area or product type.
- Tier area is determined by measures. You can select one to ten measures, such as order quantity or order price.

The following example describes how to use a butterfly chart to compare the order quantities, order prices and average profits of different products. The `company_sales_record` dataset is used in this example.

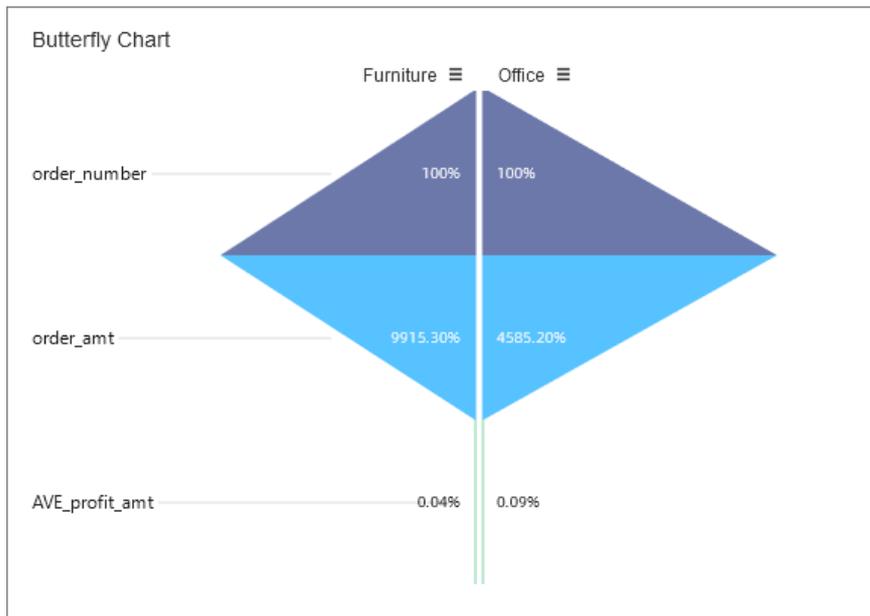
Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, click **MyDatasets** and find the `company_sales_record` dataset, then click the  icon in the **Actions** column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the  icon.
6. On the **Data** tab, select required dimensions and measures.
 - In the **Dimensions** list, find `product_type` and drag it to the **Comparison Metrics (Dim.)** field.
 - In the **Measures** list, find the `order_number`, `order_amt`, and `average profit_amt` and drag them to the **Tier Area (Mea.)** field.

Specify fields for the butterfly chart



7. Click **Update**. The chart is updated.



8. On the **Style** tab, configure parameters in the Basic Information, Chart Type, and Series Settings sections.
9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
10. Click **OK** to save the dashboard.

If you want to delete the chart, click the  icon in the upper-right corner of the chart and select **Delete**.

7.5.4.25. Create a hierarchy chart

A hierarchy chart organizes and displays hierarchical data in a tree structure. It is an implementation of the enumeration method. For example, when you view revenues of cities in a province, the relationships between the province and the cities are displayed in a hierarchical structure. Hierarchy charts are ideal for analyzing data related to organizational structures, such as the staff structure of a company or department structure of a hospital.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A hierarchy chart consists of node metrics and node labels. Node labels are determined by dimensions, such as area and product type. Node metrics are determined by measures, such as order quantity and order price.

This topic uses the following scenarios as examples to describe how to use a hierarchy chart and the filter:

- Scenario 1: Compare the order quantities of different products in the provinces of different regions.
- Scenario 2: View the average profits of different products in different municipalities.

In a hierarchy chart, you must specify at least two dimensions for node labels. Data is displayed clearly if these dimensions have a hierarchical relationship. You must specify at least one measure for node metrics.

Scenario 1: Compare the order quantities of different products in the provinces of different regions

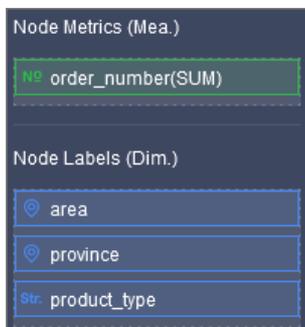
Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the Datasets page that appears, find the `company_sales_record` dataset, click the **Create Dashboard** icon in the Actions column. In the dialog box that appears, select **Standard** and click OK.
4. On the dashboard edit page, click the **Hierarchy Chart** icon.
5. Click the **Data** tab and select the required dimensions and measures.

In the Dimensions list, find and add `area`, `province`, and `product_type` to the Node Labels (Dim.) field. The sequence in which you add these dimensions determines the hierarchical relationship in the chart. In the Measurement list, find and add `order_number` to the Node Metrics (Mea.) field, as shown in the following figure.

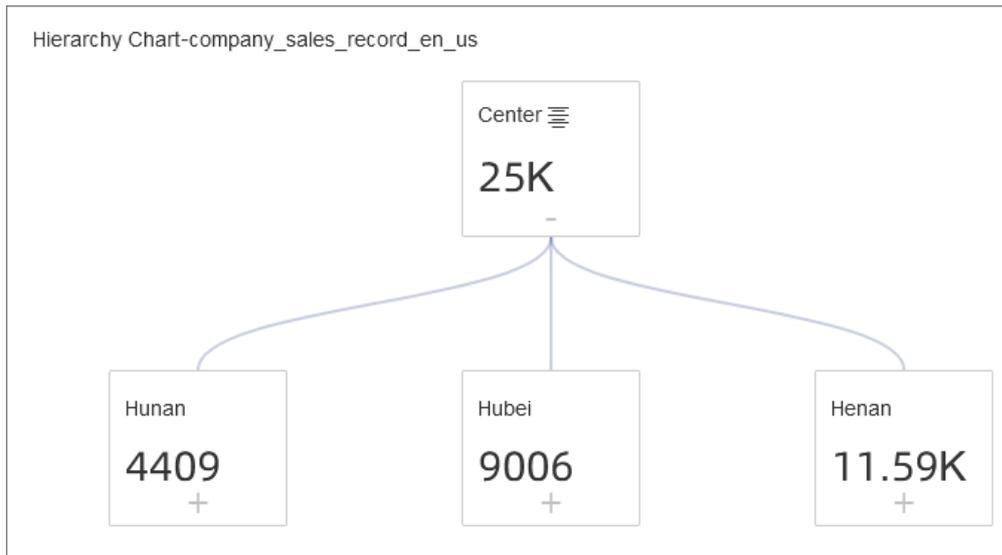
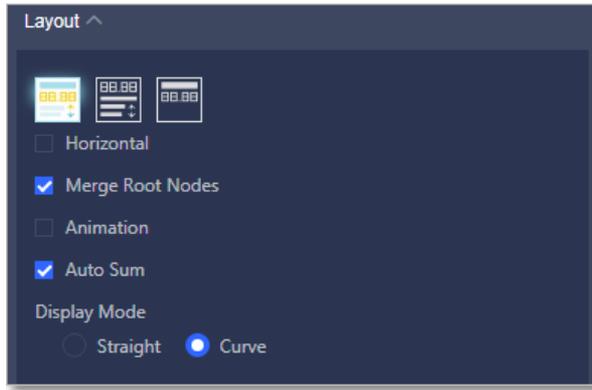
Note Ensure that you have converted the dimension type of `area` and `province` from String to Geo. For information about how to convert a dimension type, see [Edit a dimension](#).

Specify fields for the hierarchy chart



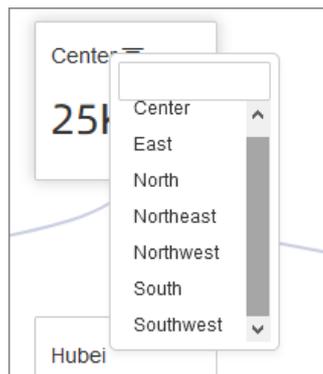
6. Click **Update**. The chart is updated.
7. Click the **Style** tab and change the title, layout, and design of the chart.
 - i. Quick BI provides three layout types for hierarchy charts. You can select a structure and display mode for the chart based on your business needs. The Merge Root Nodes check box is selected by default. In the following example, the **Straight** mode is selected.

Layout



- Move your pointer over the area field in the chart and switch to another region from the drop-down list that appears, as shown in the following figure.

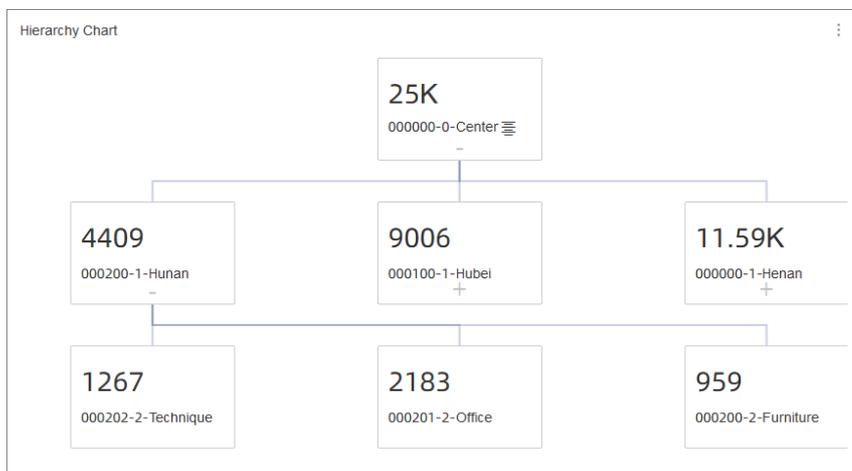
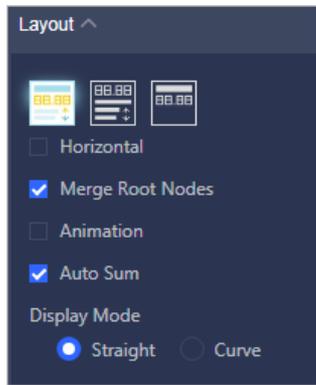
Switch to another region



- Click the minus sign (-) or plus sign (+) to fold or unfold the child nodes.

- In the Layout section, if you select **Auto Sum**, the chart displays the total amount of each node, as shown in the following figure.

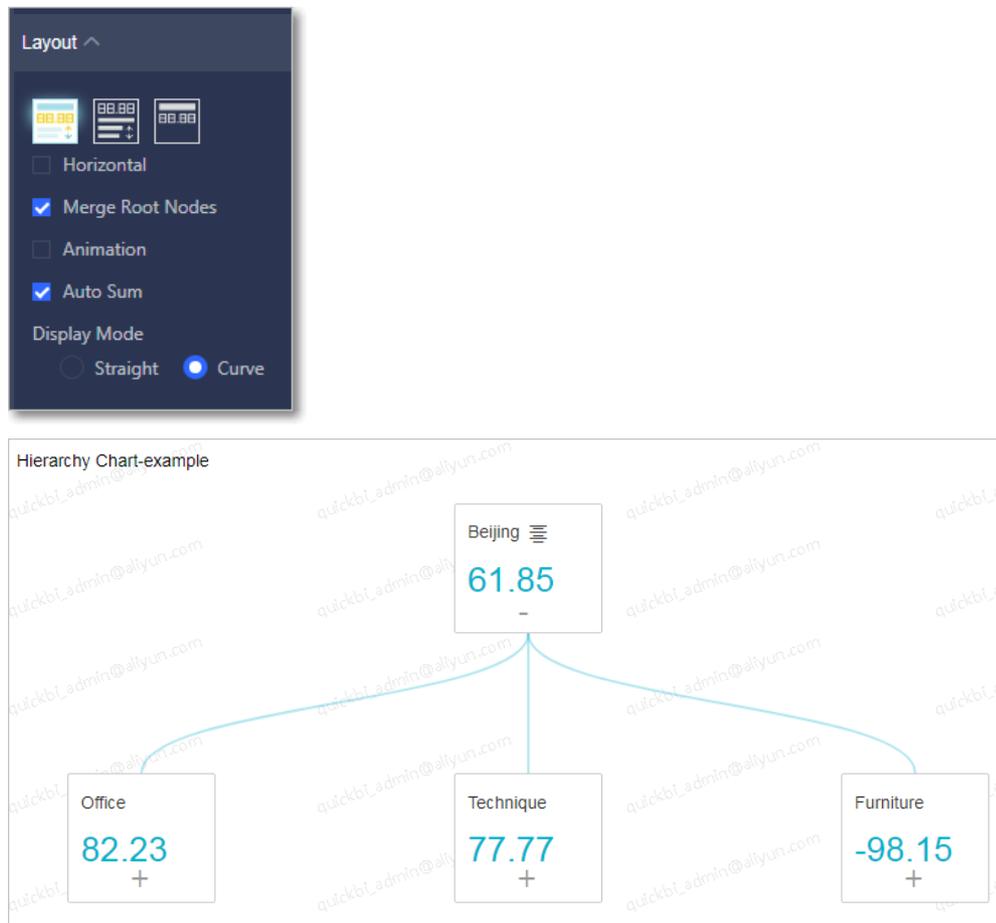
Auto Sum



- ii. In the Design section, you can configure the Levels field to specify the level of data to be displayed. You can also set a primary path, which is displayed in a different color in the chart. You can select the Show Filter Bar check box to add a toolbar to the chart. This allows you to edit the chart in preview mode or in the dashboard.

In the following example, the Primary Path parameter is set to order_number, the Sort parameter is set to Ascend, the Show Filter Bar check box is selected, and the Curve display mode is selected, as shown in the following figure.

Hierarchy chart



8. Click **Save** in the upper-right corner. In the Save Dashboard dialog box that appears, enter a name for the dashboard.
9. Click **OK** to save the dashboard.

By default, the dashboard is saved to the **My Items** tab on the Dashboards page.

Scenario 2: View the average profits of different products in different municipalities

Context

Data modeling may be required in this scenario. For information about data modeling, see [Add a calculated field](#).

Procedure

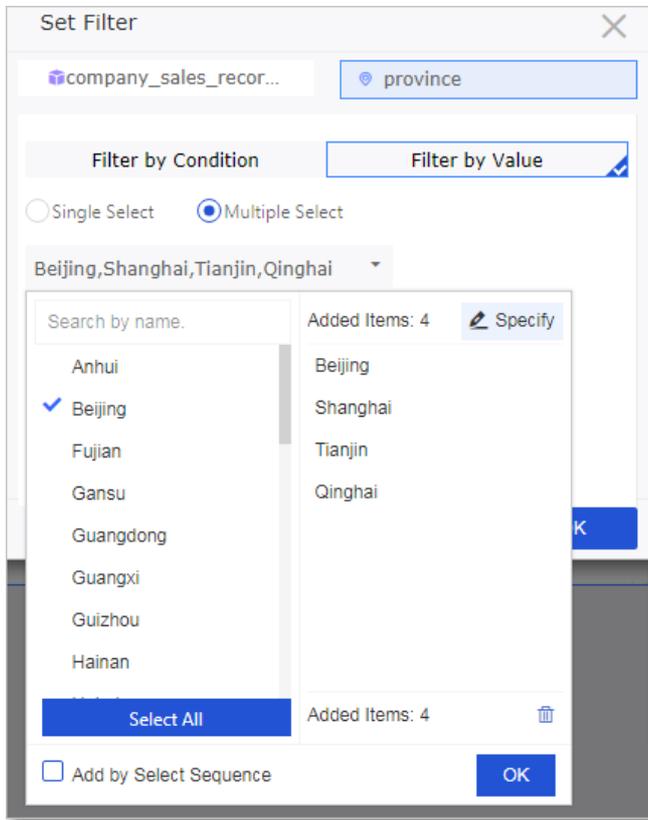
1. In the Dimensions list, find and add province to the Filters field.

This way, you can filter municipalities.

- 2. Click the **Filter** icon. In the Set Filter dialog box that appears, select **Filter by Value** and **Multiple Select**, as shown in the following figure.

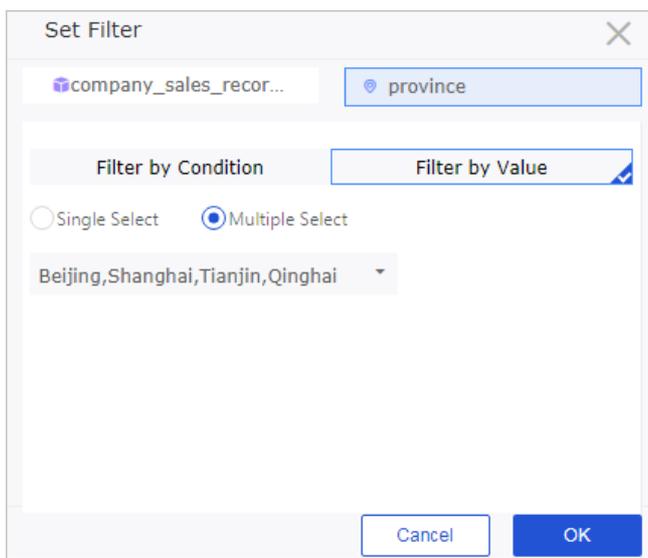
The system automatically lists all available options of the province field in the drop-down list.

Filter by Value



- 3. Select the municipalities or manually enter their names.
- 4. Click **OK** to set the filter condition, as shown in the following figure.

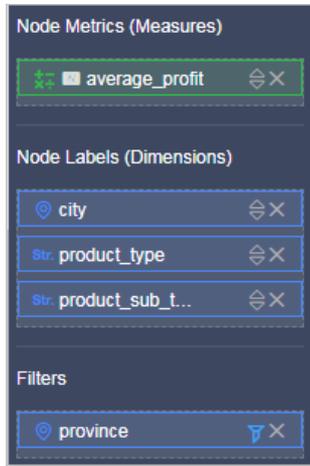
Set the filter condition



5. In the Dimensions list, find and add city, product_type, and product_sub_type to the Node Labels (Dim.) field, as shown in the following figure.

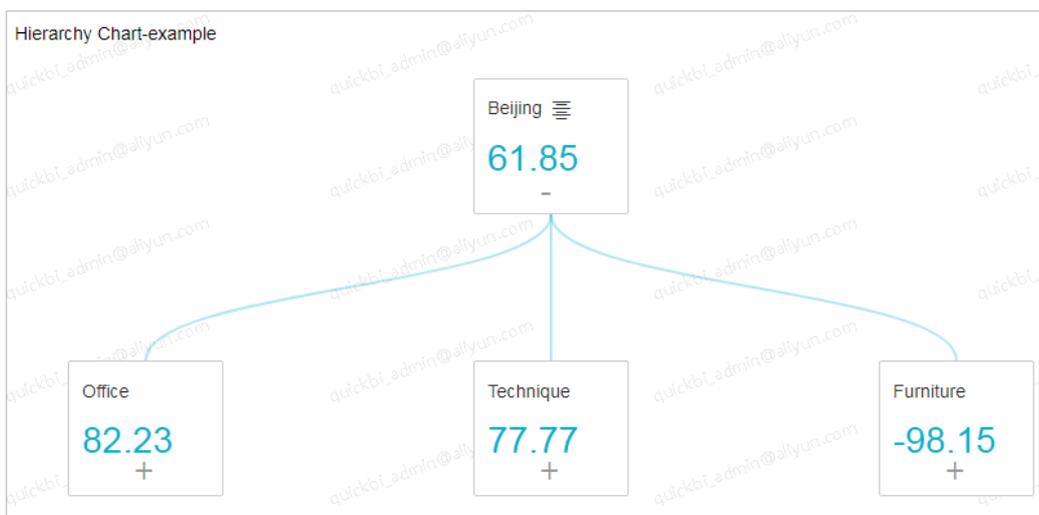
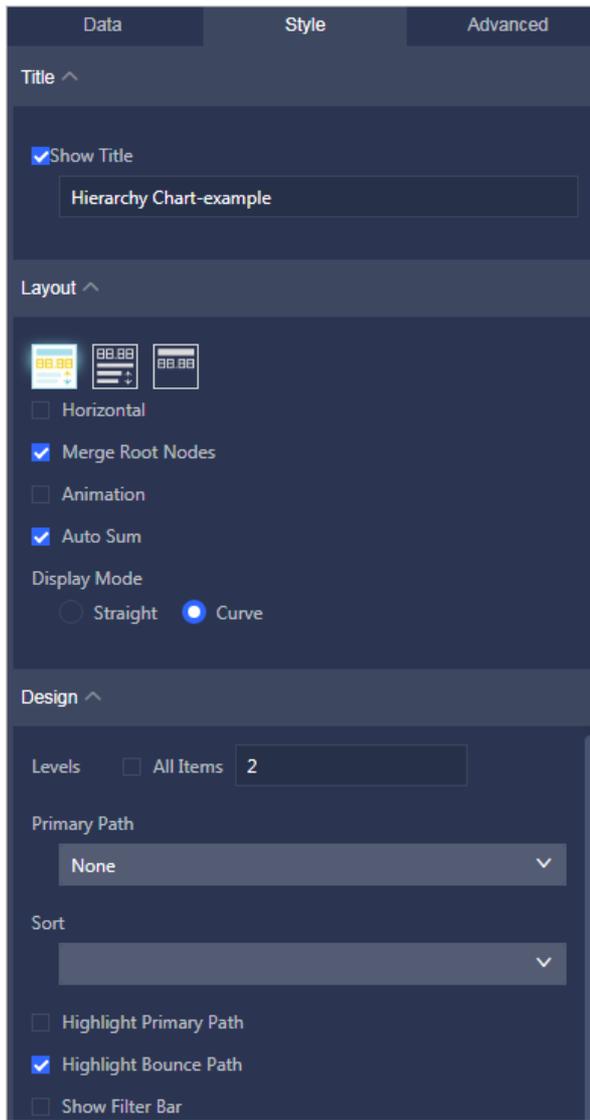
The sequence in which you add these dimensions determines the hierarchical relationship displayed in the chart. In the Measures list, find and add average_profit to the Node Metrics (Mea.) field.

Specify fields for the hierarchy chart



6. Click **Update**. The chart is updated.
7. Click the **Style** tab and change the title, layout, and design of the chart, as shown in the following figure.

Hierarchy chart



- Click **Save** in the upper-right corner. In the Save Dashboard dialog box that appears, enter a name for the dashboard.
- Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

7.5.4.26. Create a flow analysis chart

A flow analysis chart shows the data flow of source, central, and destination nodes.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

A flow analysis chart displays a data flow, including data source and destination. It also displays the proportions of data that go to different destinations. For example, assume that the total number of page views (PVs) on the Taobao homepage on a specific day is 100, you can use a flow analysis chart to analyze the number and proportion of PVs on the Taobao homepage through different search engines, as well as the number and proportion of PVs that flow to other platforms.

When you create a flow analysis chart, you must specify **Central Node (Dimensions)**, **Node Type (Dimensions)**, **Node Name (Dimensions)**, and **Node Indicator (Measures)**.

- The central node is determined by a dimension. You can select only one dimension.
- The node type is determined by a dimension. You can select only one dimension.
- Node names are determined by a dimension. You can select only one dimension.
- The node indicator is determined by a measure. You can specify only one measure.

For more information, see [Flow analysis-demo](#).

Create a flow analysis chart

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the *company_sales_record* dataset and click the  icon in the Actions column.
4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Flow Analysis** icon.
6. On the **Data** tab, select required dimensions and measure.

In the Dimensions list, find the required dimensions and add them to the Central Node (Dimensions), Node Type (Dimensions), and Node Name (Dimensions) fields. In the Measures list, find the required measure and add it to the Node Indicator (Measures) field.

7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure the title and data display format of the chart.
9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
10. Click **OK** to save the dashboard.

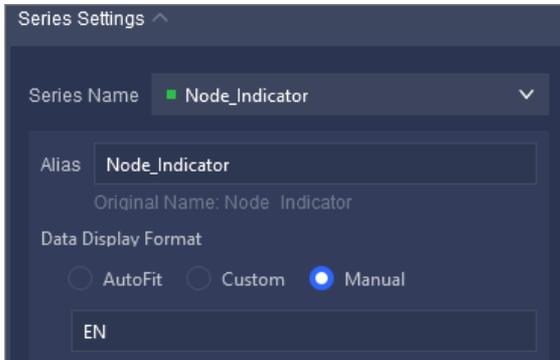
If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

Configure the parameters on the Style tab

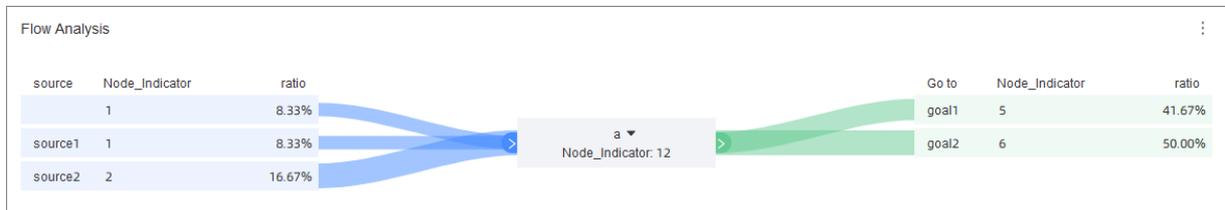
1. In the **Basic Information** section, configure Title, Description, and Show Link.

Note If you want users to be redirected to a report or an external page, select Show Link and specify Link Text and Link Address.

2. In the **Series Settings** section, configure Alias and Data Display Format for a measure.



The following figure shows an updated flow analysis chart.



7.5.4.27. Create a sankit diagram

A sankit diagram is a flow diagram in which the branch width is proportional to the flow rate. It shows the data flow between two groups of values. It is ideal for visualization analysis of energy, material composition, and finance data.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

You can specify two to five dimensions, such as province and product type, for the node type. You can specify up to one measure, such as order quantity, for the node height.

The following example describes how to use a sankit diagram to compare the order quantities and order levels of products in different regions. The company_sales_record dataset is used in this example.

Create a sankit diagram

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Dat assets**.
3. On the Datasets page, find the company_sales_record dataset, click the **Create Dashboard** icon in the Actions column. In the dialog box that appears, select **Standard** and click OK.
4. On the dashboard edit page, click the **Sankey Diagram** icon.
5. On the **Data** tab, select the required dimensions and measure.

In the Dimensions list, find area and order_level and add them to the **Node Type (Dimensions)** field.

In the Measures list, find `order_number` and add it to the **Node Height (Measures)** field.

6. Click **Update**. The chart is updated.
7. On the **Style** tab, configure parameters in the Basic Information, Chart Type, and Series Settings sections.
8. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
9. Click **OK** to save the dashboard.

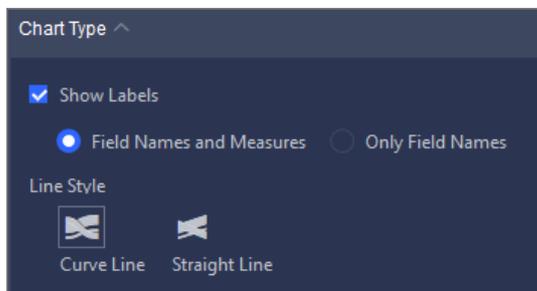
If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

Configure the parameters on the Style tab

1. In the **Basic Information** section, configure the title and whether to show the link.

 **Note** If you want to redirect to a report or an external page, select **Show Link** and specify Link Text and Link Address.

2. In the **Chart Type** section, configure Show Labels and Line Style.



3. In the **Series Settings** section, configure Display Name and Number Formatting. In this example, **AutoFit** is selected.

7.5.4.28. Create a Leaderboard

A leaderboard shows the ranking of the Top N objects from a specific measure in descending order. It objectively reflects the strengths of objects of the same category.

Prerequisites

- The Quick BI service is purchased.
- A dataset is created.

Context

You can specify only one dimension, such as area or product type, for **Category / Dimension**, and only one measure, such as order quantity or profit, for **Indicators / Measures**. A leaderboard shows 20 records by default and can show up to 50 records.

The following example describes how to use a leaderboard to compare the order quantities in different areas. The `company_sales_record` dataset is used in this example.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the **Datasets** page, find the `company_sales_record` dataset and click the  icon in the Actions

column.

4. In the **Create Dashboard** dialog box, select **Standard** or **Full Screen**, and click **OK**.
5. On the dashboard edit page, click the **Leaderboard** icon.
6. On the **Data** tab, select the required dimension and measure.
 - o In the Dimensions list, find **area** and add it to the **Category / Dimension** field.
 - o In the Measures list, find **order_number** and add it to the **Indicators / Measures** field.
7. Click **Update**. The chart is updated.
8. On the **Style** tab, configure parameters in the Basic Information, Style Settings, Functionality Settings, and Series Settings sections.
9. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, enter a name for the dashboard.
10. Click **OK** to save the dashboard.

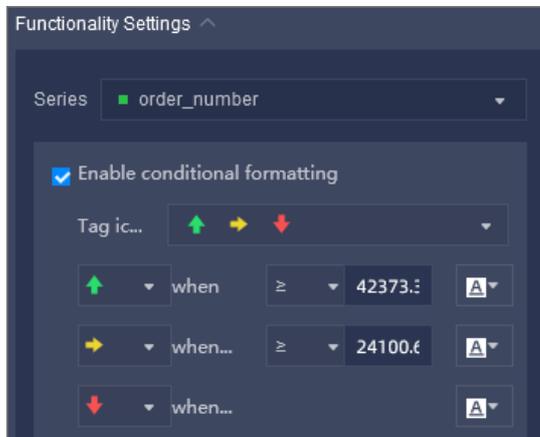
If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

Configure the parameters on the Style tab

1. In the **Basic Information** section, configure the title, color, and whether to show the link.

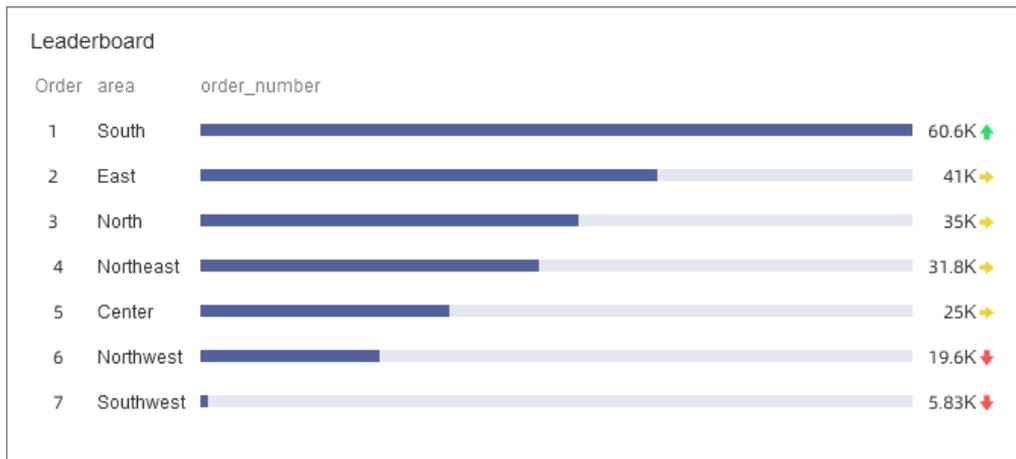
Note If you want to redirect to a report or an external page, select **Show Link** and specify Link Text and Link Address.

2. In the **Style Settings** section, configure Show Column Name, Theme for Top 3 Items, Data Display Format, Value Alignment, and Bar Color.
3. In the **Functionality Settings** section, configure conditional formatting.



- i. Select **Enable conditional formatting**, and select icon themes from the drop-down list of **Tag icon**.
 - ii. Specify the rules for marking data, the icon style, and font color.
4. In the **Series Settings** section, specify the alias for the measure or dimension, alignment mode, and data display format.

Click Update. The chart is updated, as shown in the following figure.



7.5.4.29. Create a ticker board

A ticker board displays KPI data and allows you to customize style settings such as the background color.

Prerequisites

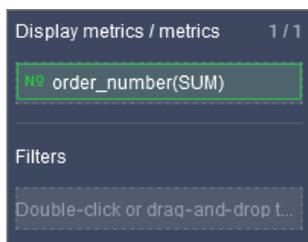
- The Quick BI service is purchased.
- A dataset is created.

Context

Ticker boards can display only metrics (measures). The following example describes how to use a ticker board to show order quantities. The `company_sales_record` dataset is used in this example.

Create a ticker board

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** tab, click **Datasets**.
3. On the Datasets page, find the `company_sales_record` dataset, click the **Create Dashboard** icon in the Actions column. In the dialog box that appears, select **Standard** and click OK.
4. On the dashboard edit page, click the **Ticker Board** icon.
5. On the **Data** tab, add the measure `order_number` to the **Indicator (Measures)** field.



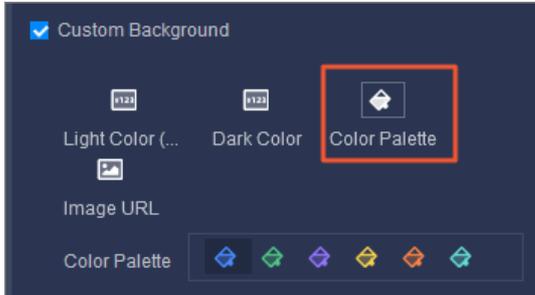
6. Click **Update**. The chart is updated.
7. On the **Style** tab, configure parameters in the Basic Information, Style Settings, Functionality Settings, and Series Settings sections.
8. Click **Save** in the upper-right corner. In the Save Dashboard dialog box, specify a name for the dashboard.
9. Click **OK** to save the dashboard.

If you want to delete the chart, click the **More** icon in the upper-right corner of the chart and select **Delete**.

Configure the parameters on the Style tab

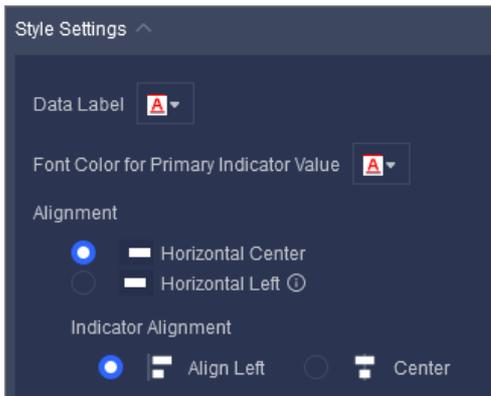
1. In the **Basic Information** section, specify Show Title and Description, Show Link, and Background.

You can select Light Color (Default), Color Palette, or Image URL for Background. Quick BI provides six color palettes.

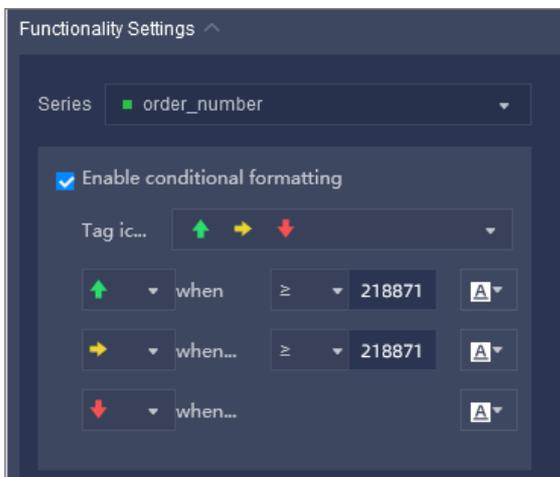


Note If you want users to be redirected to a report or an external page, select **Show Link** and specify Link Text and Link Address.

2. In the **Style Settings** section, specify the font colors for data labels and primary indicator values and the indicator position in an indicator block.

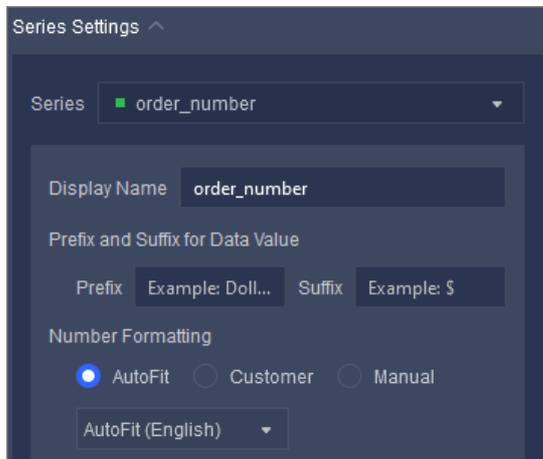


3. In the **Functionality Settings** section, configure conditional formatting.

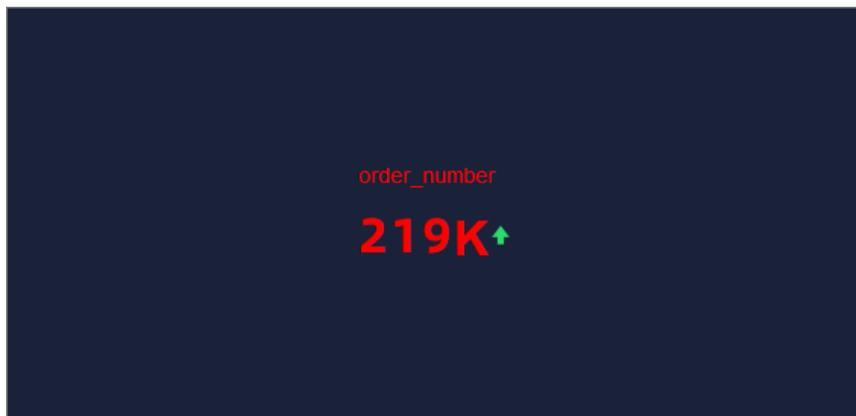


- i. Select **Enable conditional formatting**, and select icon themes from the drop-down list of **Tag icon**.

- ii. Manually enter the data value that needs to be marked, and configure the icon style and font color based on the system prompt.
4. In the **Series Settings** section, specify the measure alias, prefix, suffix, and data display format.



Click Update. The following figure shows an updated ticker board.



7.5.5. Theme design

You can use the theme design feature to change the template, skin, and theme color for a dashboard.

Prerequisites

- The Quick BI service is purchased.
- A dashboard is created. For more information, see [Go to the dashboard edit page](#).

Procedure

1. Click the  icon in the top navigation bar.
2. In the **Theme Design** section of the **Page Settings** page, configure **Official Template**, **Skin**, and **Theme Color**.

Parameter	Description
-----------	-------------

Parameter	Description
Official Template	<p>The official template is selected by default. You can change it to a custom template.</p> <p>If the templates in the drop-down list cannot meet your business requirements, click Add Custom Template to create a custom template. For more information, see Manage theme templates.</p> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>Note After you change the theme, the custom settings of the original theme cannot be restored. We recommend that you save a copy of a theme before you change it.</p> </div>
Skin	<p>Two options are provided: Light and Dark.</p> <ul style="list-style-type: none"> If you select Light, the background color of the dashboard is white. The default value is Light. If you select Dark, the background color of the dashboard is black.
Theme Color	<p>Three options are provided: Business, Classic, and Custom.</p> <p>If you select Custom for Theme Color, you can select a color system from the drop-down list.</p> <div style="border: 1px solid #333; padding: 10px; margin-top: 10px; background-color: #1a2b4d; color: white;"> <p>Theme Design</p> <p>Official Template Orange</p> <p>Skin</p> <p><input type="checkbox"/> Light <input type="checkbox"/> Dark</p> <p>Theme Color</p> <p>Business Classic Summer solstice is not</p> <p>Select Orange</p> <ul style="list-style-type: none"> Orange Data Blue female fashion Calm senior Rouge blush natural forest national tide retro </div> <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p>Note After you change the theme color, the custom settings of the original theme cannot be restored. We recommend that you save a copy of a theme before you change it.</p> </div>

7.5.6. Full screen mode

This topic describes the basic features of the full screen mode.

The full screen mode allows you to perform the following operations in the display area of a dashboard:

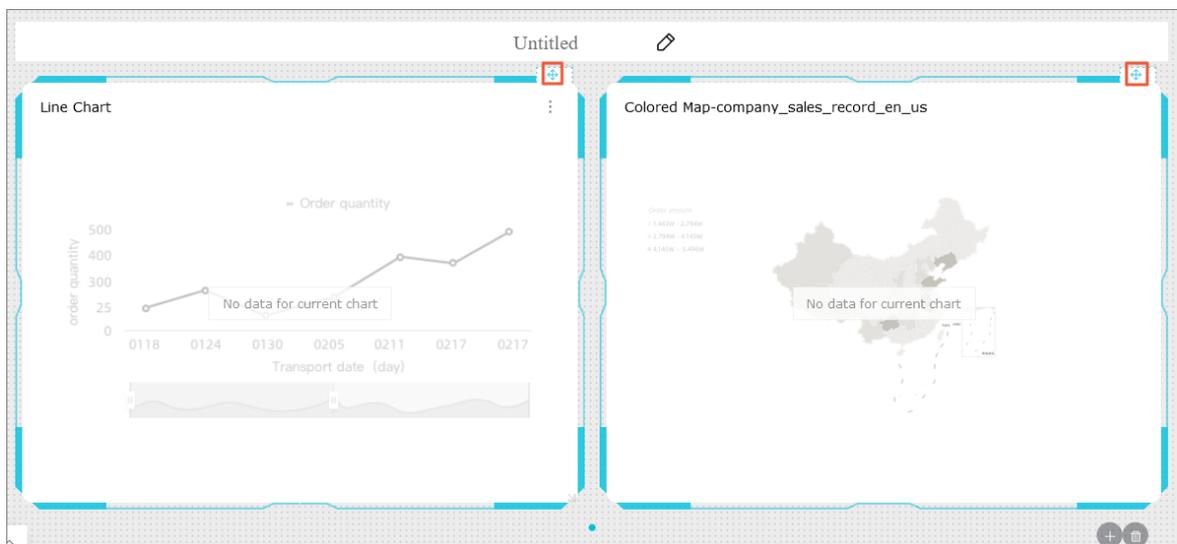
- Adjust chart positions

- Add a subscreen
- View chart data
- Delete a chart
- Change chart types
- Configure page settings

Adjust chart positions

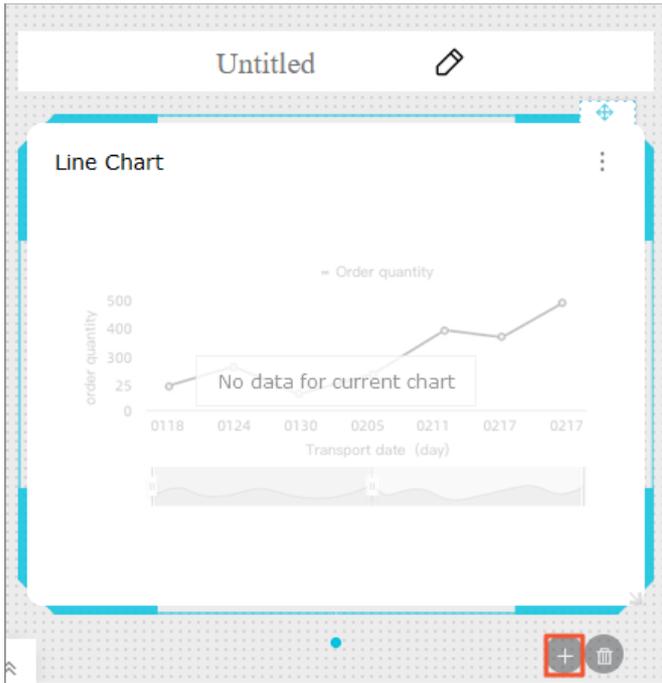
In full screen mode, a single chart can be displayed in the entire display area of a dashboard. If you have created multiple charts, you can perform the following operations to adjust their positions:

1. [Access a dashboard](#).
2. In full screen mode, you can click the cross-arrow icon in the upper-right corner of the chart to drag the chart to the specified position.

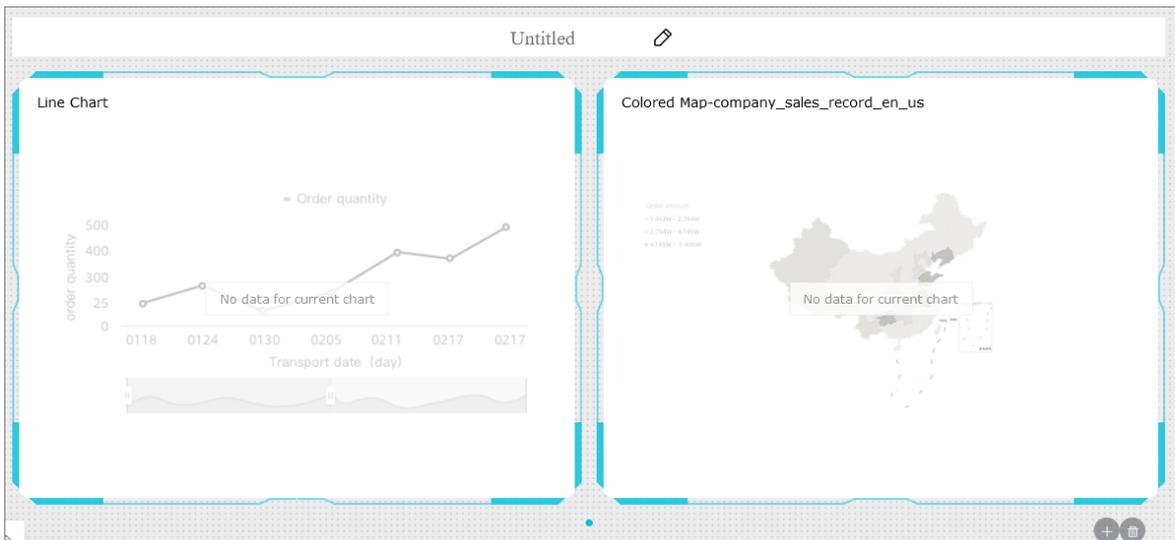


Add a subscreen

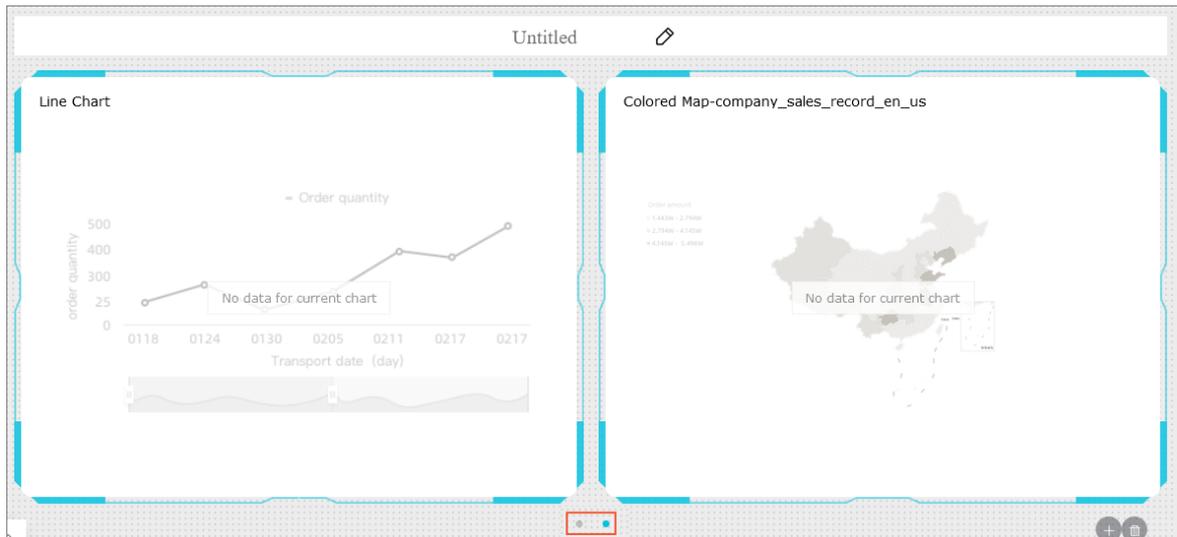
1. Click the plus sign in the lower-right corner.



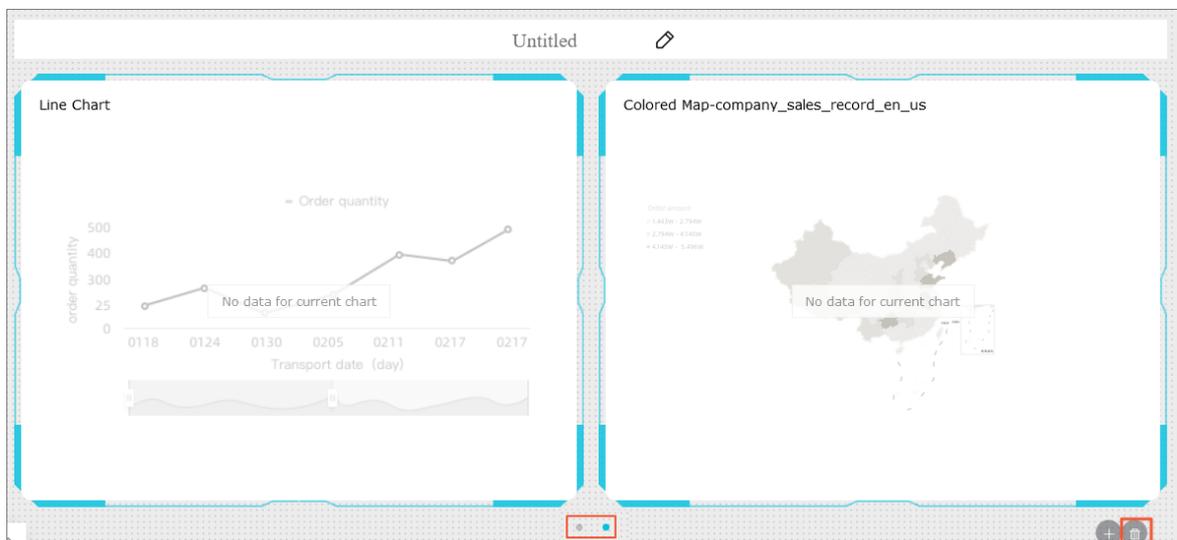
2. Add charts to the subsection.



3. Click Switch Screen to switch from the current screen to another.



4. Click the Delete icon in the lower-right corner to delete a subscreen.



Move a chart

1. Select a chart.
2. Click the  more icon in the upper-right corner of the chart.
3. Select **Move To**.

You can move the chart to a containers tab.

View chart data

1. Select a chart.
2. Click the  more icon in the upper-right corner of the chart.
3. Select **View Data**.

You can view the data of the chart.

View SQL statements

1. Select a chart.
2. Click the  more icon in the upper-right corner of the chart.
3. Select **View SQL Statements**.

You can view the SQL statements that corresponds to the chart.

View full screen

1. Select a chart.
2. Click the  more icon in the upper-right corner of the chart.
3. Select **Full Screen**.

The chart can be displayed in full screen.

Delete a chart

1. Select a chart.
2. Click the  more icon in the upper-right corner of the chart.
3. Select **Delete**.

You can delete the chart.

Change chart types

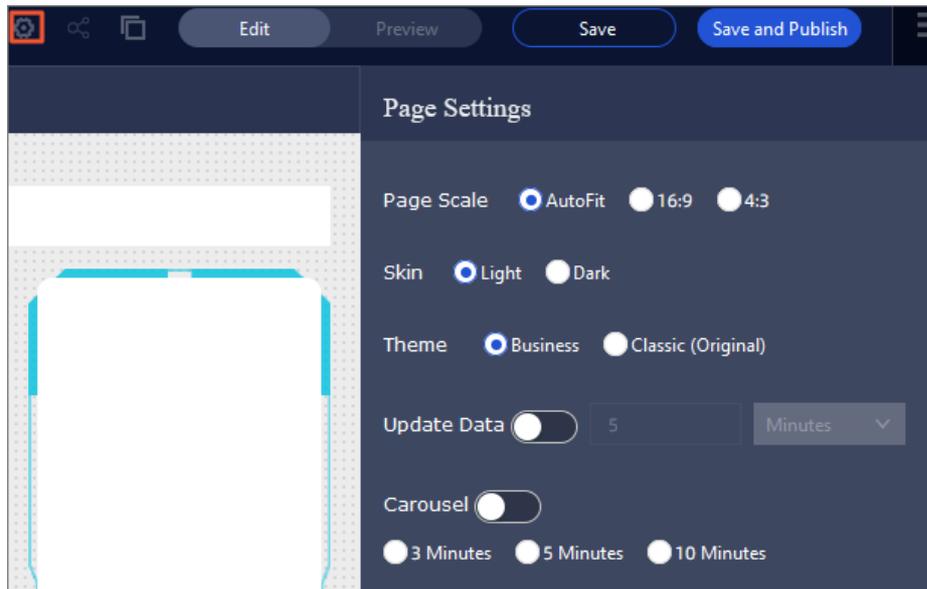
1. Select a chart.
2. In the Graphic Design area, click **Change Chart Type**.
3. Click the chart type you want.

Note

If the fields of the current chart do not match those of the target chart, the chart type cannot be changed. You must manually adjust fields before you change the chart type. The system provides instructions to help you adjust the elements based on the current and target chart types. To change the chart type, follow the instructions to adjust the dimensions and measures.

Page settings

You can click the  icon to view the page settings and specify the page scale, skin color, theme, data update interval, and time interval of data carousel.



7.5.7. Search for a dashboard

This topic describes how to search for a specific dashboard.

Prerequisites

- The Quick BI service is purchased.
- One or more dashboards are created.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. Enter a keyword in the search box to search for the required dashboard.

7.5.8. Create a dashboard folder

This topic describes how to create a dashboard folder.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, click **Create Folder** in the upper-right corner.
4. In the **Create Folder** dialog box, specify a name for the folder and click **OK**.

7.5.9. Rename a dashboard folder

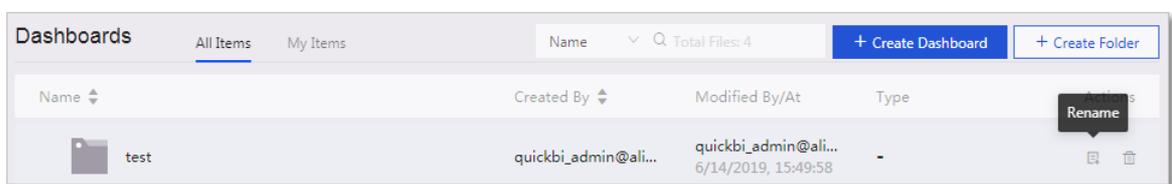
This topic describes how to rename a dashboard folder.

Prerequisites

- The Quick BI service is purchased.
- A dashboard folder is created. For information about how to create a dashboard folder, see [Create a dashboard folder](#).

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, find the dashboard folder you want to rename.
4. Click the  icon.



5. In the Rename dialog box, enter a name and click **OK**.

7.5.10. View a dashboard

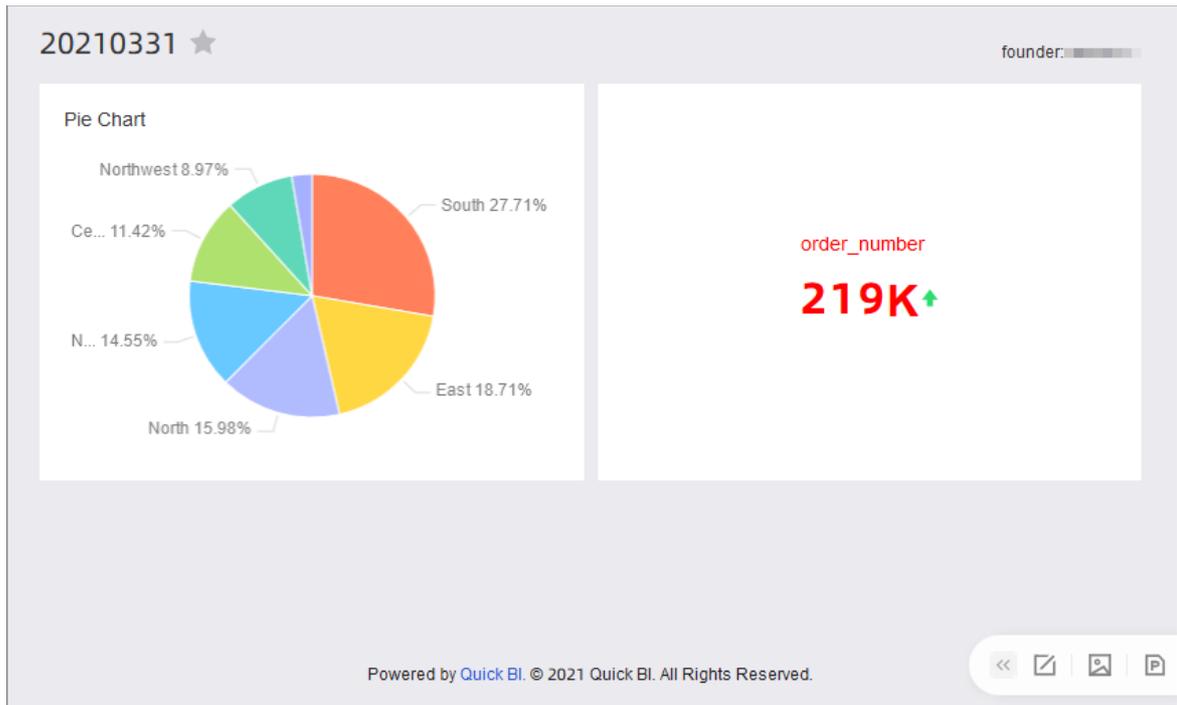
This topic describes how to view a dashboard.

Prerequisites

- The Quick BI service is purchased.
- One or more dashboards are created.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, find the required dashboard and click the  icon next to the dashboard. On the page that appears, you can view the information of the dashboard.



4. You can perform one of the following operations as required:
 - Edit: Click the  icon to go to the dashboard edit page to edit a chart.
 - Export: Click the  icon to export the chart data to a PNG file.
 - Export PDF: Click the  icon to export the chart data to a PDF file.

7.5.11. Share a dashboard

This topic describes how to share a dashboard with other users.

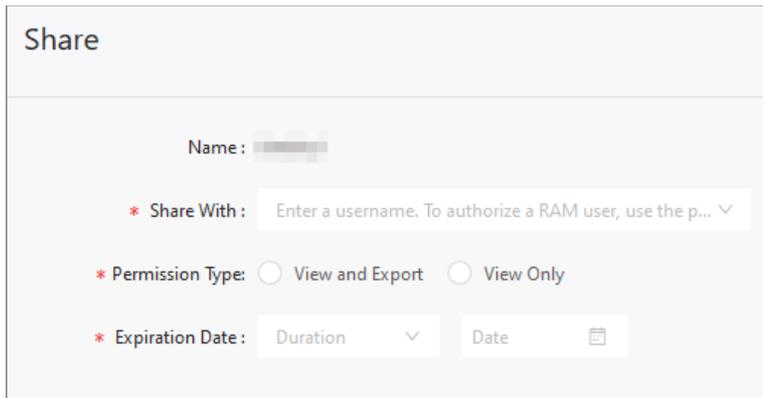
Prerequisites

- The Quick BI service is purchased.
- A dashboard is created.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, right-click the dashboard that you want to share with others.
4. Select **Share**.
5. In the right-side panel that appears, specify Scope, Permission Type, and Expiration Date, as shown in [Share a dashboard](#).

Share a dashboard



 **Note**

- In the Basic Settings section of the right-side Page Settings panel, select **Allow Download**. If you want to download a report, you must set Permission Type to **View and Export**.
- You can set Scope to All Users, User Groups, or Users.
- Users who are authorized in one of these scopes are granted the corresponding permissions.

6. Click **Save**.

7.5.12. Make a dashboard public

After you make a dashboard public, users can access the dashboard by using a shared link.

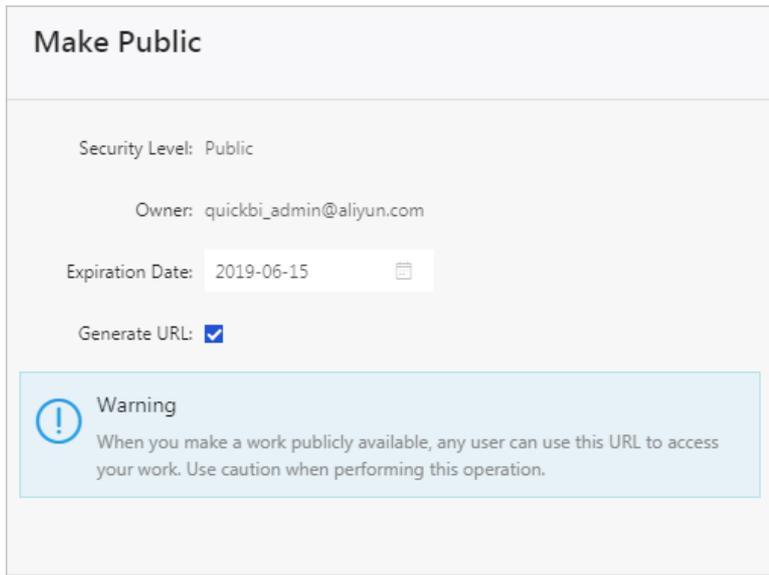
Prerequisites

- The Quick BI service is purchased.
- A dashboard folder is created. For information about how to create a dashboard folder, see [Create a dashboard folder](#).

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, right-click the dashboard you want to make public.
4. Select **Make Public**.
5. Set an expiration date and click **Make Public**, as shown in [Make a dashboard public](#).

Make a dashboard public



7.5.13. Publish a dashboard

This topic describes how to save, save and publish, restore, unpublish, and republish a dashboard.

Prerequisites

- The Quick BI service is purchased.
- A dashboard is created.

Context

Dashboards are automatically published after they are saved. If developers save modifications to the dashboards, users may view the intermediate versions that are generated. To improve user experience, the save feature is separated from the publish feature, and the unpublish and republish features are added.

Go to the dashboard configuration page

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Dashboards**.
3. On the Dashboards page, click the dashboard name to go to the dashboard editing page.

Save a dashboard

You can save only the current operation.

If you save only a dashboard, the updates to the dashboard are invisible to users:

- When you add a new dashboard, the dashboard is in the unpublished state after you save it. You can click **Save and Publish** to publish the dashboard.
- When a published dashboard is updated, you can publish the update point by clicking **Republish**.

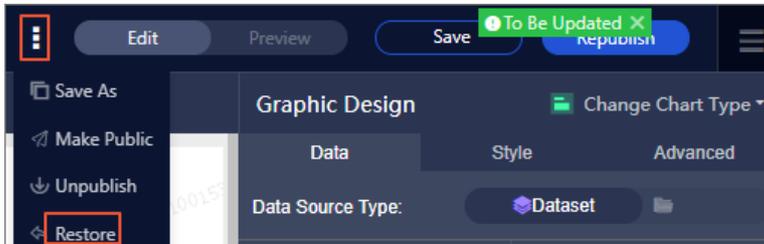
Save and publish

The save and publish feature saves the current operation and publishes the dashboard. This feature is available only for dashboards that are not published.

Restore icon

After you save your modifications to a published dashboard but have not published the modifications, you can use the **Restore** feature to restore the dashboards to the latest version.

1. Select a published dashboard.
2. Modify the dashboard and save the modifications.
The dashboard is in the **Saved But Not Published** state.
3. On the dashboard edit page, click the  icon and select **Restore**.

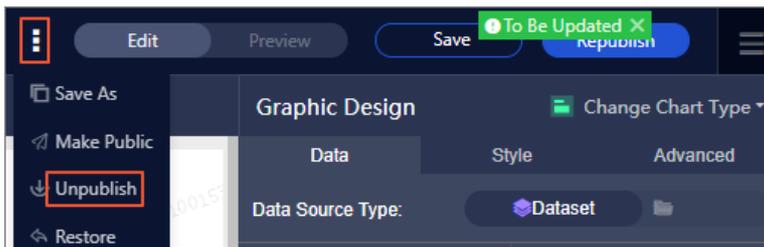


4. In the dialog box that appears, click OK.

Unpublish a dashboard

You can unpublish a dashboard. Unpublished dashboards are invisible to users.

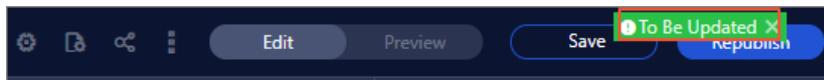
1. On the dashboard edit page, click the  icon and select **Unpublish**.



2. In the dialog box that appears, click OK.

Republish a dashboard

After you update a published dashboard, you can republish it to make the updates take effect. If "To Be Updated" is displayed for a dashboard and the dashboard is in the **Saved But Not Published** state, you can republish the dashboard.



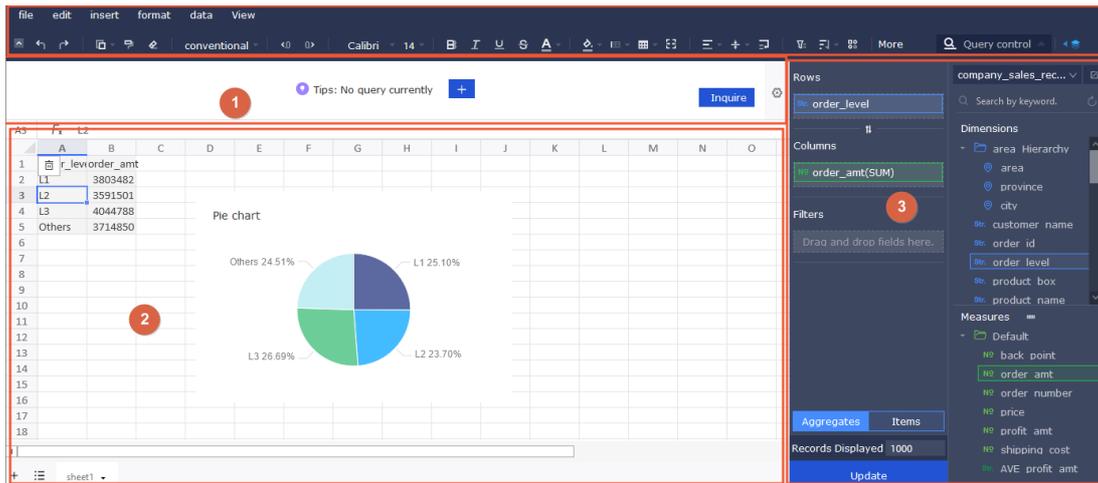
7.6. Workbooks

7.6.1. Overview

On the workbook edit page, you can filter and query data in a dataset. You can also visualize data by using different types of charts.

The workbook edit page contains three areas, as shown in [Workbook edit page](#).

Workbook edit page



- The workbook configuration section is marked as ①. In this section, you can select the chart that you want to create on the workbook, export and edit the workbook, configure the data format and style of the workbook, and add a query control.
- The workbook display section is marked as ②. In this section, you can display data in specified cells in a specified chart and reprocess data by using cell references.
- The dataset selection section is marked as ③. In this section, you can switch the current dataset to another. The fields of a dataset are displayed in the Dimensions and Measures lists based on the data types preconfigured in the system. You can select dimensions and measures based on the chart type.

7.6.2. Create a workbook

This topic describes how to create a workbook.

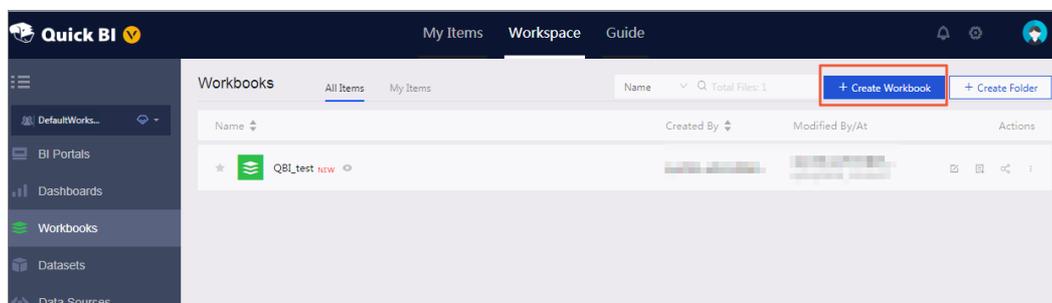
Prerequisites

The Quick BI service is purchased.

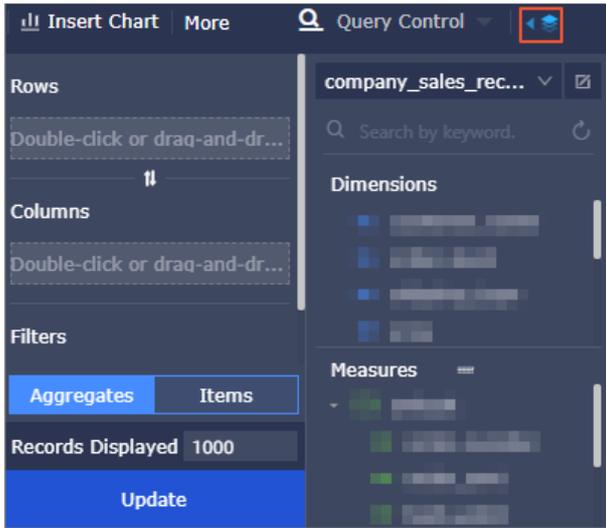
Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks**.
3. On the **Workbooks** page, click **Create Workbook**.

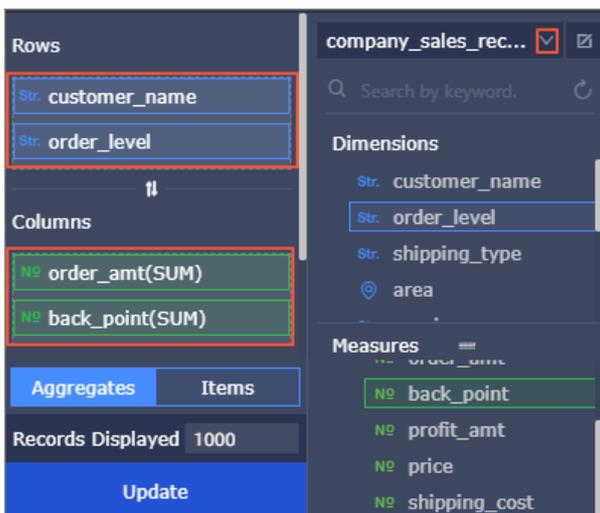
Create a workbook



4. On the workbook edit page, click the icon.



5. Click **Select a dataset** and select the required dataset from the drop-down list. Fields in the required dataset are synchronized to the **Dimensions** and **Measures** areas.
6. In the **Dimensions** and **Measures** lists, double-click or drag the corresponding fields to **Rows** and **Columns**.



7. Click **Update**.
8. Click **Save**.
9. (Optional) If it is the first time that you save the workbook, the **Save Workbook** dialog box appears. You must configure the following parameters in the dialog box. Enter a name for the workbook and specify the location where you want to store the workbook, and click **OK**.

7.6.3. Switch to another dataset

This topic describes how to select a dataset or switch from the current dataset to another dataset.

Prerequisites

The Quick BI service is purchased.

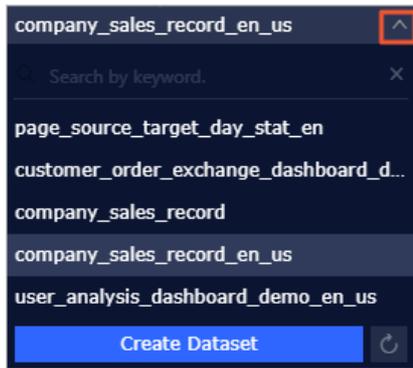
Context

If you cannot find the required dataset, go back to the dataset management page and make sure that the dataset has been created. For information about how to create a dataset, see [Create datasets](#).

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. Click the name of the required Workbook.
4. On the workbook edit page, click the downward arrow. In the drop-down list, select or search for the dataset you want to switch to, as shown in [Switch to another dataset.](#)

Switch to another dataset



7.6.4. Search for a dimension or measure

This topic describes how to search for a specific dimension or measure.

Prerequisites

The Quick BI service is purchased.

Context

After you select a dataset, the system displays the dimensions in the Dimensions list and measures in the Measures list.

For information about how to edit dimensions and measures, see [Edit a dimension](#) and [Edit a measure](#).

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. Click the name of the required Workbook.
4. On the workbook edit page, manually enter the keyword of the field in the search box in the dataset selection area. Click the search icon to search for the field.

Search for a field



7.6.5. Configure a workbook

You can configure a workbook in the workbook configuration section.

Prerequisites

The Quick BI service is purchased.

Go to the page of workbook configuration

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. On the Workbooks page, click the workbook that you want to set the font.

File icon

You can choose **File > Export** to export a workbook.

Edit icon

You can perform the following edit operations:

- Revoke
- Restore
- Copy
- Cut
- Paste
- Format Painter
- Clear Format
- Indent
- Unindent

Insert

You can perform the following insert operations:

- Insert rows or columns to a workbook.

- Insert row above
- Insert row below
- Insert column on the left
- Insert column on the right
- Insert a dataset to a workbook.
- Insert a chart to a workbook.

You can insert the following types of charts: line chart, bar chart, pie chart, dashboard, radar chart, scatter plot, funnel chart, and polar diagram. For more information, see [Insert a chart](#).

- Insert a picture to a workbook.

The supported image formats include PNG, JPG, and GIF.

- Insert a function to a workbook.
- Insert a hyperlink to a workbook.
- Insert a note to a workbook.
- Insert a drop-down box to a workbook.

For more information, see [Add a drop-down box](#).

Format

Operation	Description
Data Format	Set the Data format. The options include conventional, date, digital, text, percentage, and customize.
Font and Font Size	Set the font and font size.
Bold, Italic, Underline and Strikethrough	Configure the text style.
Horizontal alignment, Vertical alignment, Word wrap, Merge cells, Set line height, Set column width, Adaptive line height, and Adaptive column width	Configure the cell style.
Conditional Formatting	Add rules to filter data results. For more information, see Conditional Formatting .

Data

You can perform the following operations on the data in a workbook:

- Sort
- Filter
- Data Lookup
- Group and Ungroup

View

You can perform the following operations on the data in a workbook:

- Hide or display gridlines
- Hide selected rows or columns
- Unhide
- Freeze to the current line or column

- Unfreeze

Toolbar

You can perform the following operations on the data in a workbook.

Operation	Description
	Paste data with custom formatting.
	Copy all of the formatting from one object and apply it to another one.
	Clear the formatting.
	Decrease or increase the number of decimal places.
	Specify the font color.
	Specify the fill color for cells.
	Specify whether to show cell borders and the border color.
	Specify the theme color of the workbook.
	Search for data.
	Specify the theme style of a table.
	Create a query control

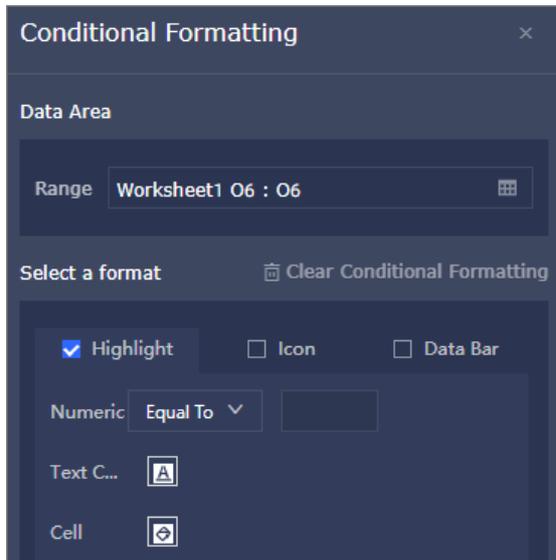
Sheet settings

You can perform the following operations on a sheet in a workbook:

- Rename the sheet.
- Delete the sheet.
- Set the border color.
- Hide the sheet.

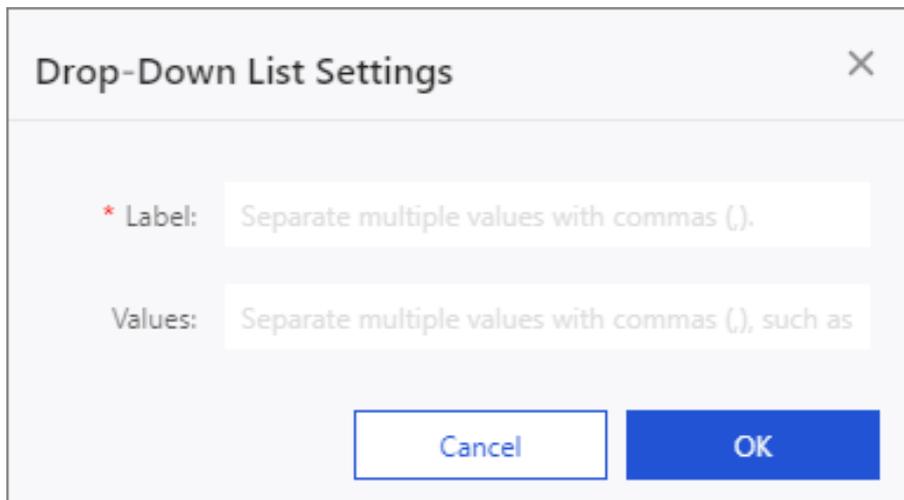
Conditional Formatting

You can click  Conditional Formatting in the toolbar and add rules to filter the data results, such as highlighting the data within a specified range, setting an icon for specified data, and adding a data bar.



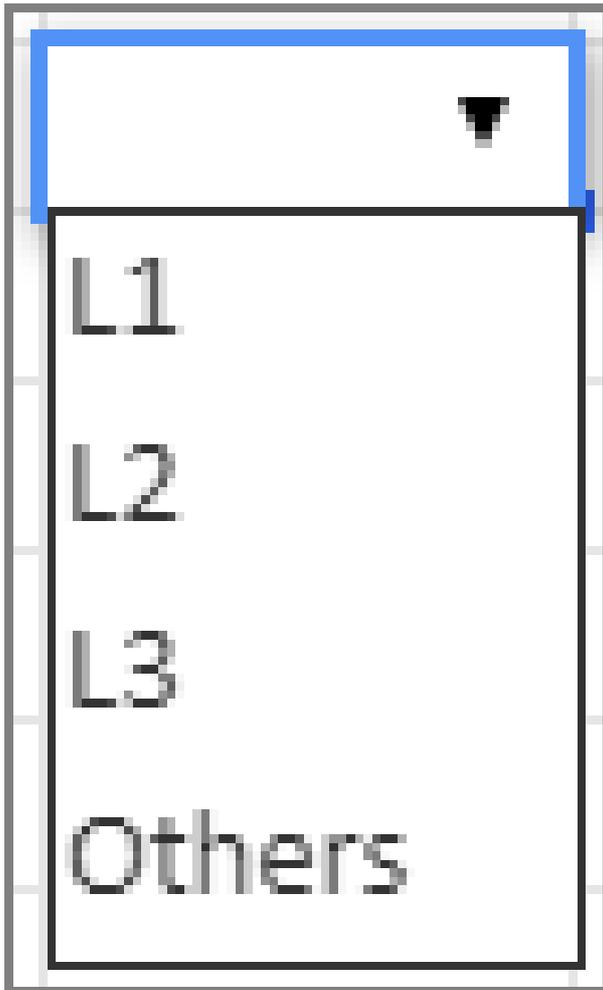
Add a drop-down box

1. In the toolbar on the workbook edit page, choose **Insert > Drop-Down Box**.
2. In the Drop-Down Box Settings dialog box, add a label for the data item.



 **Note** Separate multiple labels with commas (,).

3. Click **OK**.

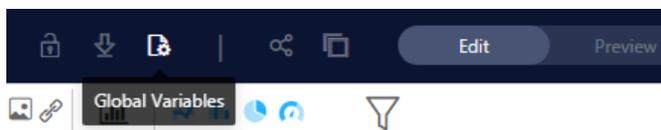


Note If the message **This operation is not allowed because it may affect the data of nearby datasets.** appears, copy the data to another area in the workbook and then repeat the operations.

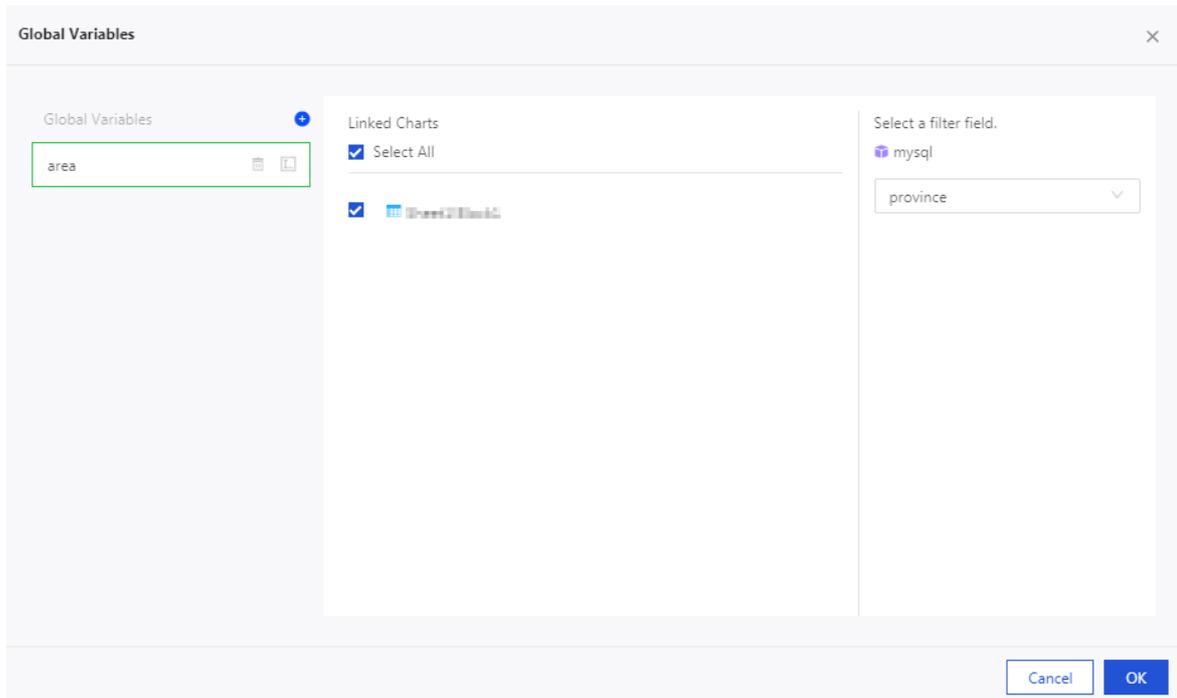
Global Variables

You can set global variables in the workbook. You can use global variables to add hyperlinks to cross tables.

1. On the workbook edit page, click the **Global Variables** icon.



2. In the **Global Variables** dialog box, enter the required variable and click **OK**, as shown in the following figure.



Insert a chart

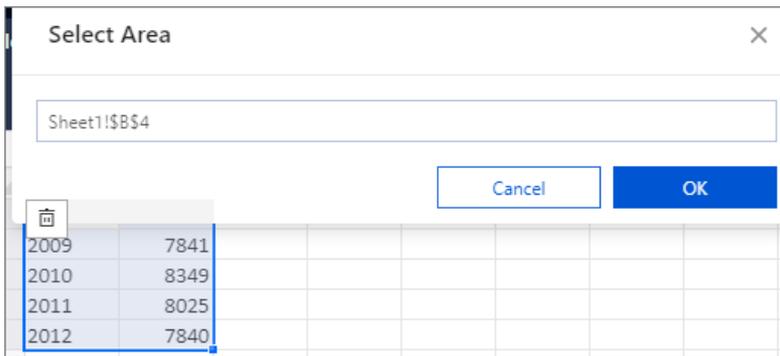
You can insert charts to a workbook based on the data in the workbook. You can insert the following types of charts: line chart, bar chart, pie chart, dashboard, radar chart, scatter chart, funnel chart, and polar diagram.

1. In the toolbar of the workbook edit page, click **Insert** and select **Chart**, and then select a chart type.

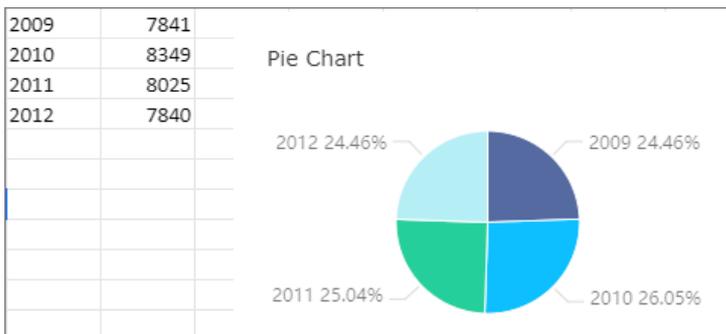
In this example, Pie Chart is selected.



- In the **Select Data Range** dialog box, specify the data range for creating the chart.



- Click **Determine** . A pie chart appears in the workbook.



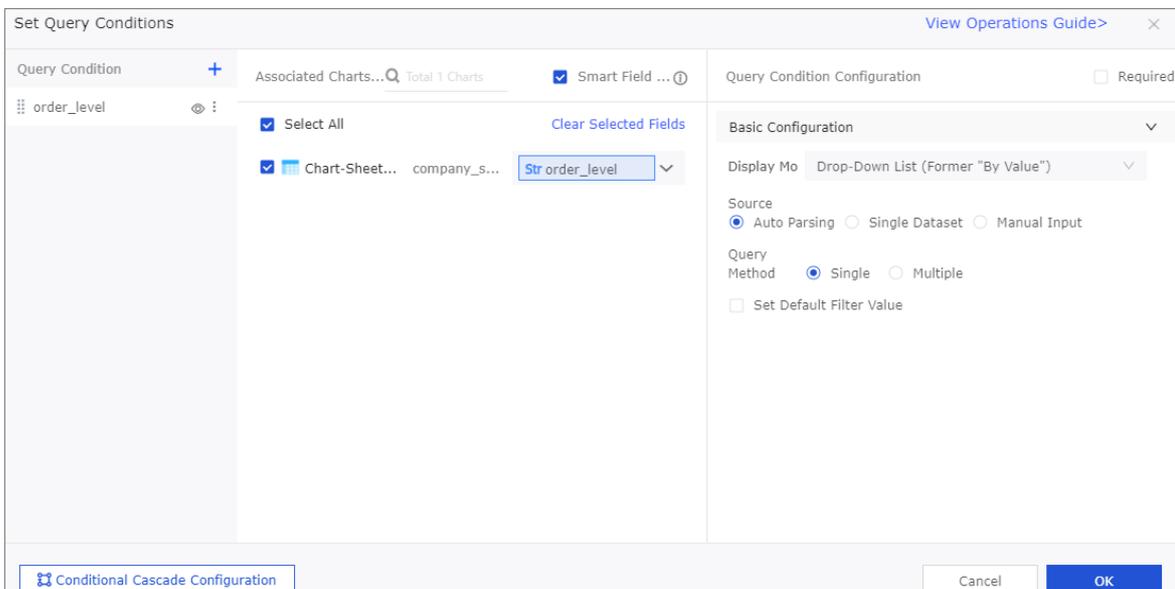
Click the chart and configure the chart style in the **Graphic Design** section on the right side.

Create a query control

- On the workbook edit page, click **Query Control** in the toolbar.

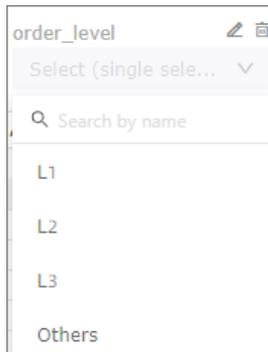


- In the workbook display section, click the **+** icon.
- In the **Query condition setting** dialog box, configure query conditions.



4. Click **OK**.

You can view the order level on the Query Control page.



7.6.6. Manage a workbook

This topic describes how to search, move, view, share, transfer, and delete a workbook. This topic also describes how to add a workbook to favorites and make a workbook public.

Prerequisites

The Quick BI service is purchased.

Search for a workbook

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. On the **Workbooks** page, enter the keywords or the creator of the workbook that you want to search for in the search box, and click the **Search** icon.

Move a workbook

1. On the **Workbooks** page, find the workbook that you want to move, click the  icon in the **Actions** column, and then select **Move To**.

You can also right-click the workbook and select **Move To**.

2. On the **Move To** page, select the required directory and click **Save**.

Share a workbook

Only workbooks whose **Publish Status** is **Published** can be shared with others.

1. On the **Workbooks** page, find the workbook that you want to share with others and click the  icon in the **Actions** column.

You can also right-click the workbook and select **Share**.

2. In the right-side panel that appears, specify Scope, Permission Type, and Expiration Date.

 **Note** You can set Scope to All Users, User Groups, or Users.

3. Click **Save**.

Make a workbook public

Only workbooks whose **Publish Status** is **Published** can be made public.

1. On the **Workbooks** page, find the workbook that you want to make public, click the  icon in the **Actions** column, and then select **Make Public**.

You can also right-click the workbook and select **Make Public**.

2. In the **Make Public** panel, specify **Expiration Date** and select **Generate URL**.
3. Click **Make Public**.

Transfer a workbook, rename a workbook, and change its security level

1. On the **Workbooks** page, find the required workbook and click the  icon in the **Actions** column.

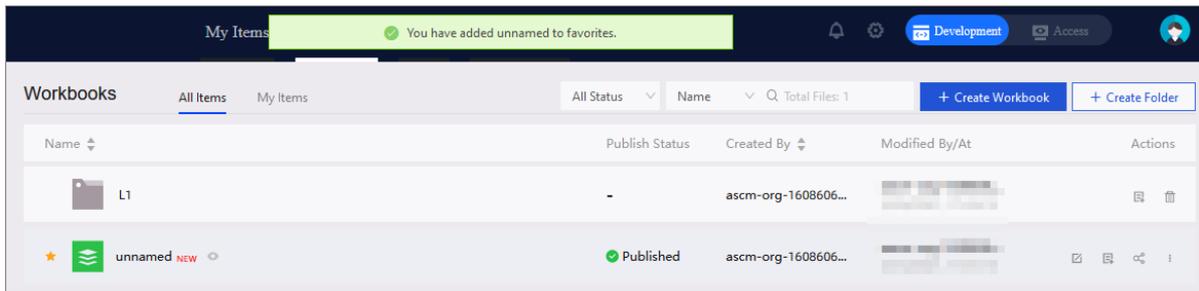
You can also right-click the workbook and select **Edit Properties**.

2. In the **Edit Properties** panel, change the name, owner, description, and security level of the workbook.
3. Click **Save**.

Add a workbook to favorites

You can use any of the following methods to add a workbook to favorites:

- On the **Workbooks** page, click the  icon next to the workbook name.



- On the workbook edit page, click the  icon next to the workbook name.



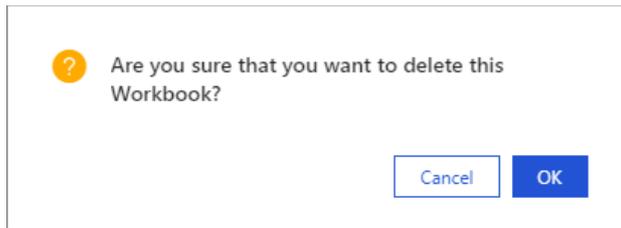
 **Note** You can click the  icon again to remove the workbook from your favorites.

Delete a workbook

1. On the **Workbooks** page, find the workbook that you want to delete, click the  icon in the **Actions** column, and then select **Delete**.

You can also right-click the workbook and select **Delete**.

2. In the message that appears, click **OK**.



Note When you delete a workbook in the **Published** state, you need to unpublish it before deleting it.

7.6.7. Publish a workbook

This topic describes how to save, save and publish, restore, unpublish, and republish a workbook.

Prerequisites

The Quick BI service is purchased.

Context

Workbooks are published automatically after they are saved. Users may view intermediate versions that are generated as developers save changes to the workbooks. To improve user experience, the save feature is separated from the publish feature, and the unpublish and republish features are added.

Go to the workbook editing page

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. On the Workbooks page, click the workbook that you want to edit.

Save a workbook

You can save only the current operation by using the save feature.

If you save only a workbook, the update points of the workbook are invisible to users:

- When you add a new workbook, the workbook is in the unpublished state after you save the workbook. You can publish the workbook by clicking **Save and Publish**.
- When a published workbook is updated, you can publish the update point by clicking **Republish**.

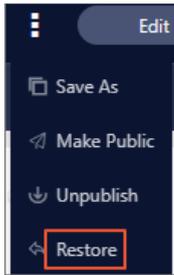
Save and publish

You can use this feature to save the current operation and publish the workbook. This feature is available only for workbooks that are not published.

Restore icon

If you save your modifications to a published workbook but have not published the modifications, you can use the **restore** feature to restore the workbook to the latest published version.

1. Select a published workbook.
2. Modify the workbook and save the modifications.
The workbook is in the **Saved But Not Published** state.
3. On the workbook edit page, click the  icon and select **Restore**.

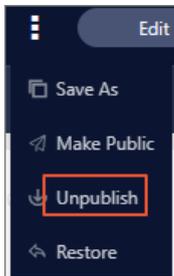


4. In the Confirm dialog box, click **OK**.

Unpublish a workbook

You can unpublish a workbook. After the workbook is unpublished, it is invisible to users.

1. Select a published workbook.
2. On the workbook edit page, click the  icon and select **Unpublish**.



3. In the dialog box that appears, click **OK**.

Republish a workbook

After you update a published workbook, you can republish it to make the updates available for users. You can republish a workbook if "To Be Updated" is displayed and the status of the workbook is **Saved But Not Published**.



7.6.8. Create a workbook folder

This topic describes how to create a workbook folder that helps you manage workbooks.

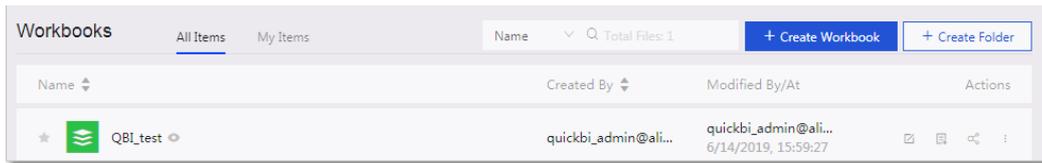
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. Click **Create Folder**, as shown in [Create Folder](#).

Create Folder



4. In the Create Folder dialog box, specify a name for the folder and click **OK**.

7.6.9. Rename a workbook folder

This topic describes how to rename a workbook folder.

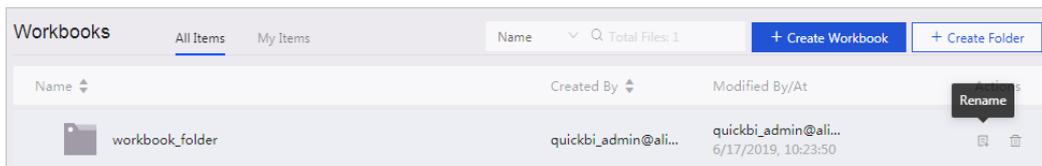
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. Find the workbook folder you want to rename and click the  icon in the Actions column, as shown in [Rename a folder.](#)

Rename a folder



4. In the **Rename** dialog box, enter a new folder name and click **OK**.

7.6.10. Delete a workbook folder

This topic describes how to delete a workbook folder.

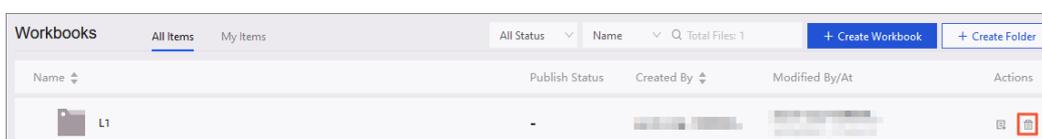
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the **Workspace** page, click **Workbooks** to go to the **Workbooks** page.
3. Find the workbook folder you want to rename and click the  icon in the Actions column, as shown in [Delete a dataset folder.](#)

Delete a dataset folder



4. In the dialog box that appears, click **OK**.

7.7. BI portals

7.7.1. Overview

A BI portal is a collection of dashboards, workbooks, and external links organized with menus. You can create a BI portal to perform complex thematic analysis with navigation panes.

7.7.2. Create a BI portal

This topic describes how to create a BI portal.

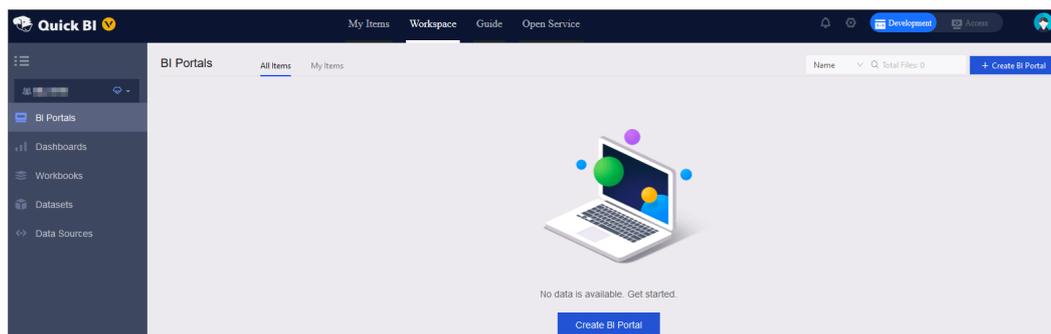
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **BI Portals**.
3. On the BI Portals page, click **Create BI Portal** in the upper-right corner, as shown in [Create a BI portal](#).

Create a BI portal



4. On the Page Settings page, configure the parameters and click **Save** in the upper-right corner.

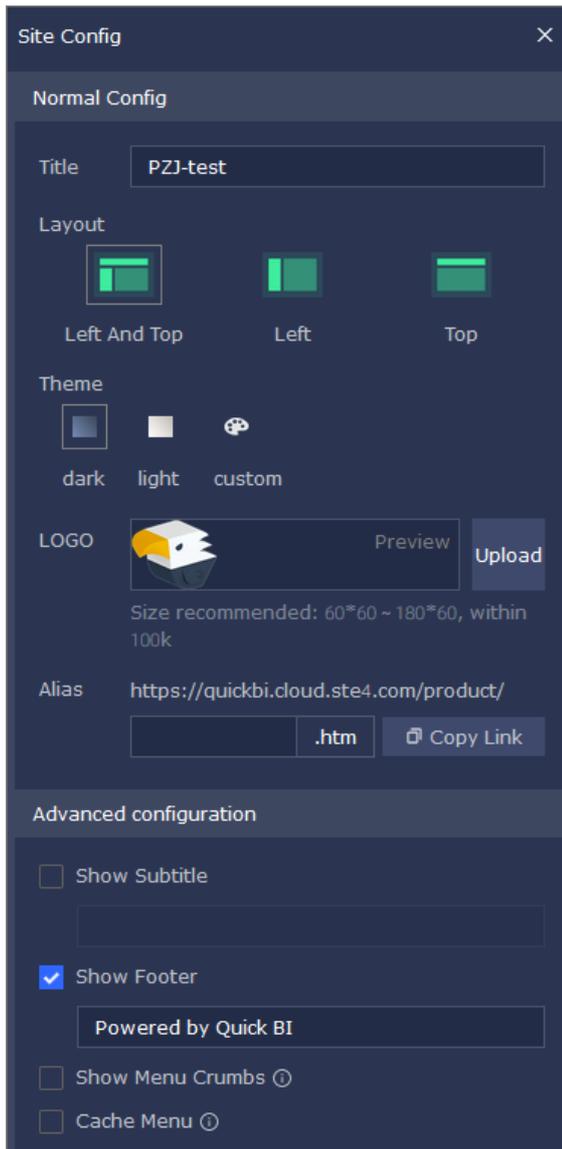
7.7.3. Configure a portal

This topic describes how to configure the title, upload the logo, and edit the footer for a BI portal on the Portal Settings page.

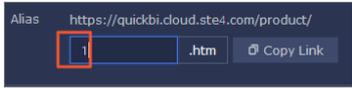
Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **BI Portals**.
3. On the BI Portals page, click the required BI portal. For more information about how to create a BI portal, see [Create a BI portal](#).
4. Click the  icon in the top navigation bar to edit the BI portal page, as shown in [Configure a BI portal](#).

Configure a BI portal



Category	Configuration item	Description
Normal Config	Title	Customize the title of the BI portal.
	Layout	You can select Left And Top, Left, or Top.
	Theme	You can select dark, light, or custom.
	LOGO	Click Upload to upload a logo image file.

Category	Configuration item	Description
	Alias	<p>Enter a value in the text box under Alias. The value is embedded in the portal link. For example, if you enter 1, the alias is <code>http://daily.yunbi.biz.aliyun.test/product/1.htm</code>.</p> 
Advanced configuration	Show Subtitle	After you select this option, add a subtitle for the portal.
	Show Footer	After you select this option, you can customize the footer, which is displayed at the bottom of the content page.
	Show Menu Crumbs	After you select this option, the menu levels are displayed at the top of the content page.
	Cache Menu	After you open multiple menus at the same time, reloading is not required when you switch between menus.

5. Click **Save**.

7.7.4. Configure menus

On the Menu Config page, you can configure the title, links, and other information of a menu.

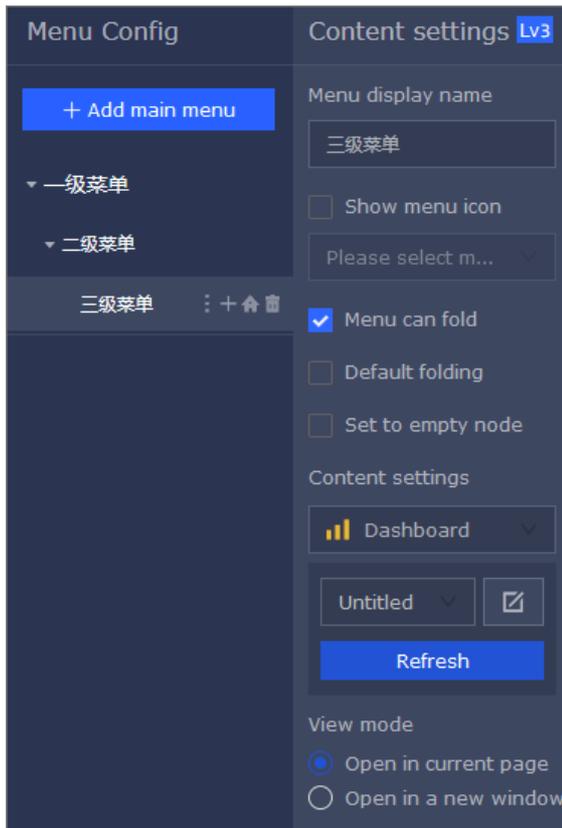
Prerequisites

A BI portal is created. For more information, see [Create a BI portal](#).

Procedure

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **BI Portals**.
3. On the BI Portals page, click the name of a BI portal.
4. In the left-side navigation pane, click the required menu, and edit the menu on the right-side **Content settings** tab, as shown in the following figure.

Configure a menu



Tab	Configuration item	Description
Menu Config		Adds a level-1 menu. Note You can also drag and drop a menu to adjust its level.
		Hides a menu. If a parent menu is hidden, its child menus are also hidden. If you want to show a menu after it is hidden, click this icon again.
		Adds a level-2 or lower menu.
		Specifies a menu as the homepage. After you select this option, the homepage is displayed with priority when you open the BI portal.

Tab	Configuration item	Description
		Deletes a menu.
Content settings	Menu display name	Specifies the name of a menu.
	Show Menu Icon	Sets the icon for a menu.
	Menu can fold	Specifies whether a menu can be collapsed.
	Default folding	Specifies the default status of a menu. Note This option can be selected only when Allow Menu Collapse is selected.
	Set to empty node	If this option is selected, you cannot set the menu content.
	Content settings	The following content types are supported: <ul style="list-style-type: none"> ◦ Dashboard ◦ Excel ◦ External Link
	View mode	Open in current page and Open in a new window are supported. Note This field cannot be configured if Set to empty node is selected.

5. Click **Save**. The configured menu appears on the left side of the current page.

7.7.5. Manage BI portals

You can share your BI portals with other users. You can also edit the properties of a BI portal. Specifically, you can rename the BI portal, transfer it to another user, or change its security level.

Prerequisites

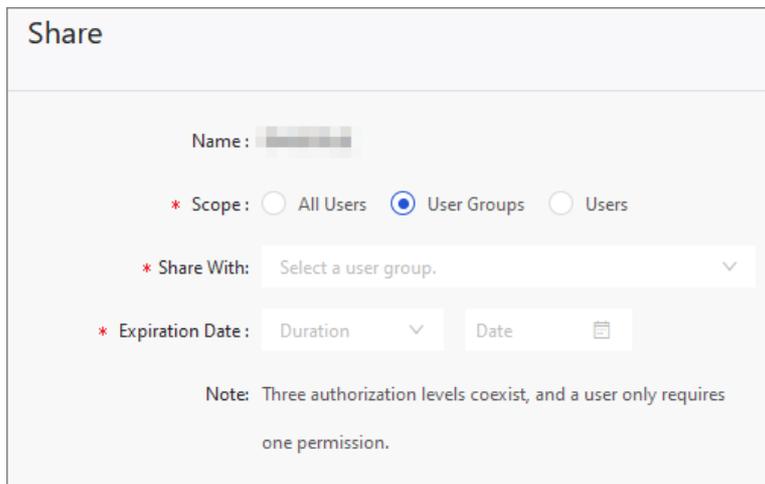
A BI portal is created. For more information, see [Create a BI portal](#).

Share a BI portal

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **BI Portals**.
3. On the BI Portals page, find the required BI portal, and click the  icon in the **Actions** column. You can

also right-click the required BI portal and select **Share**.

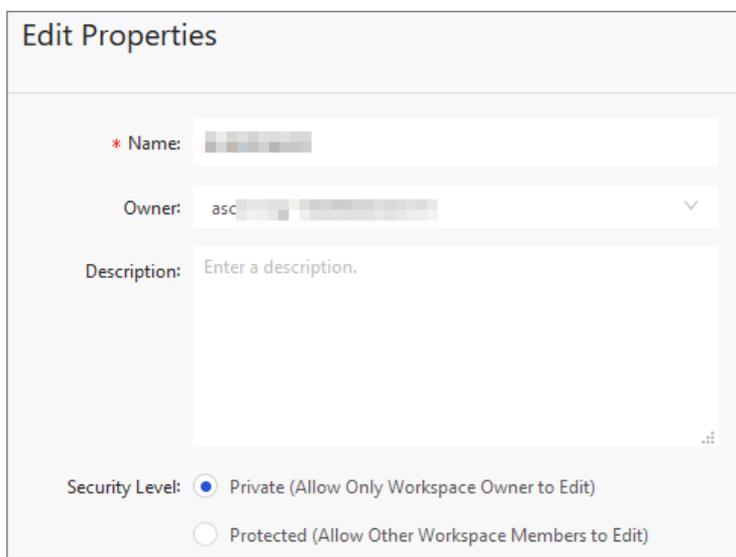
- Specify the authorization scope and validity period.



- Click **Save**.

Rename a BI portal, transfer it to another user, or change its security level

- On the BI Portals page, find the required BI portal, and click the  icon in the **Actions** column. You can also right-click the required BI portal and select **Edit Properties**.
- In the **Edit Properties** panel, you can rename the BI portal, transfer it to another user in the same workspace, or change its security level. If the security level is set to **Protected (Allow Other Workspace Members to Edit)**, the lock mechanism is triggered when multiple users attempt to edit the BI portal at the same time.



 **Note** You can transfer a BI portal to another user or set the security level of a BI portal only in group workspaces.

7.7.6. Manage menu permissions of a BI portal

Workspace administrators can manage menu permissions of BI portals.

Prerequisites

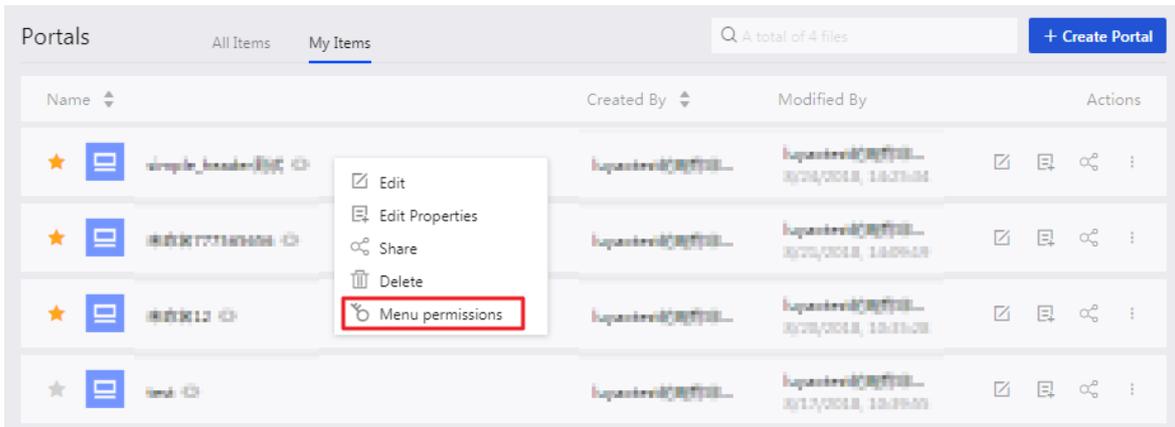
A workspace is created. For more information, see [Create a workspace](#).

Procedure

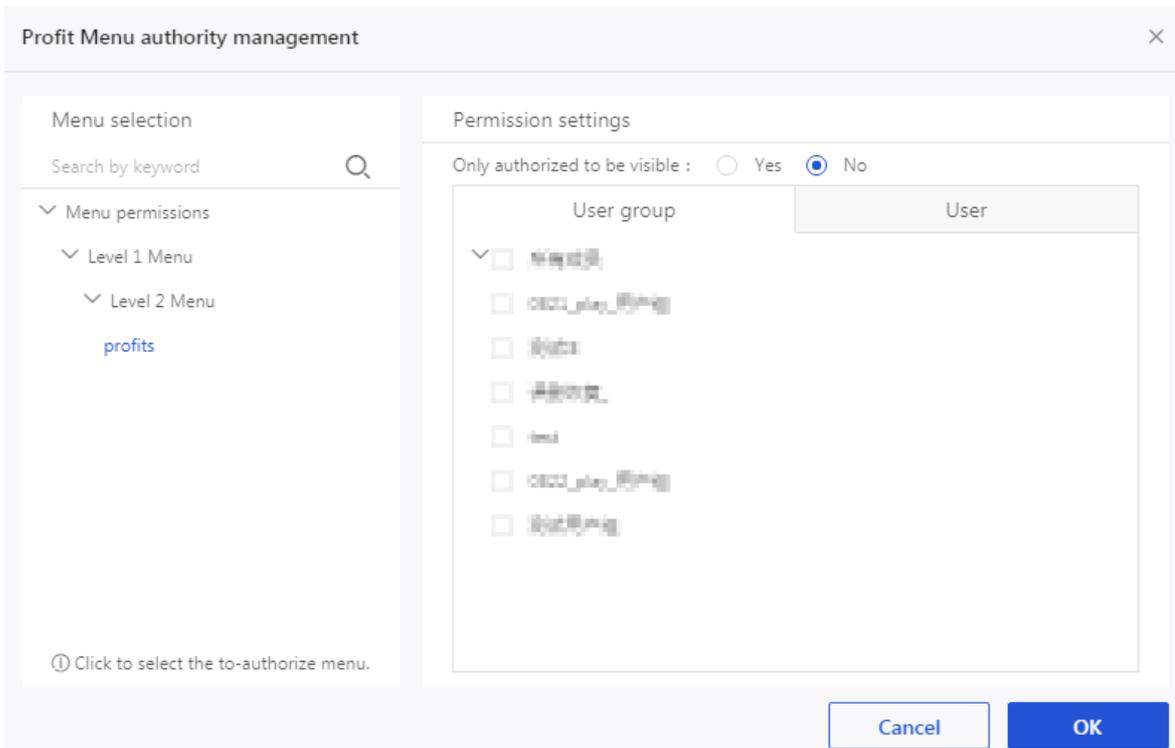
Workspace administrators can grant menu permissions to a specified user or user group.

1. [Log on to the Quick BI console](#).
2. In the left-side navigation pane of the **Workspace** tab, click **BI Portals**.
3. On the BI Portals page, find the required BI portal, click the  icon in the **Actions** column, and then select **Manage Menu Permissions**.

You can also right-click the required BI portal and select **Manage Menu Permissions**.



4. In the **Manage Menu Permissions of BI portal** dialog box, select a menu item for authorization, specify whether the menu is available only to authorized users, and select the users or user groups that you want to authorize. The configuration takes effect immediately.



Note This configuration takes effect only for users who have the BI portal view permissions. If the users and user groups are selected in the Manage Menu Permissions dialog box:

- If you select **Yes for Available Only to Authorized Users**, only the authorized users and user groups have permissions to view the menu.
- If you select **No for Available Only to Authorized Users**, all user groups and users have permissions to view the menu.

After you select users and user groups, they are authorized to view the menu content, such as the dashboard, regardless of whether the BI portal is shared.

For more information about how to create a user group, see [Create a user group](#).

5. Click **Close** to complete menu permission management.

7.8. Organization

7.8.1. Overview

An organization typically refers to a small or medium enterprise, public institution, college department, or a department of a large enterprise.

If your organization has a large number of members, requires multiple members to collaborate on data analysis, and has high requirements on data security, Quick BI provides the following features to meet your requirements:

- Different departments have access to different reports.
- Members with different roles have access to different data.

Members in an organization are classified into two types: administrators and common members.

7.8.2. Create an organization

This topic describes how to create an organization.

Prerequisites

Before you create an organization, you must create an Apsara Stack tenant account in the Apsara Stack console. Each Apsara Stack tenant account can be used to create or join in only one organization. Make sure that you have not created or joined an organization before.

Procedure

1. [Log on to the Quick BI console](#).
2. On the homepage of the **Quick BI** console, or in a workspace, click the **Settings** icon.



3. Click **Organization** in the left-side navigation pane.
4. Select the **Agree** check box and click **Create Organization**.
5. In the **Create Organization** dialog box, enter an organization name.

7.8.3. Modify organization information

Quick BI allows you to modify the information about an organization as needed.

Prerequisites

The Quick BI service is purchased.

Context

Administrators of an organization have the permissions to modify the information about the organization.

The screenshot shows the 'Organization Basics' configuration page. It features a header with 'Organization' and sub-tabs: 'Basics', 'Members', 'User Groups', and 'AccessKey'. Below the header is a section titled 'Organization Information'. The main content area contains several form fields: 'Name' (with a placeholder and a detailed validation rule), 'Description' (containing 'QuickBI testubg'), 'Created At' (showing '3/14/2019, 10:26:47'), and 'Owner' (with a placeholder). At the bottom of the form are two buttons: 'Save' and 'Leave Organization'.

Administrators of an organization have the permissions to add members to the organization to collaborate on tasks.

Administrators of workspaces have the permissions to add members to their workspaces based on the roles and responsibilities of the members. You can create workspaces that correspond to the departments of the organization. For example, if an organization has a sales department and an HR department, administrators can create a workspace for each of the two departments, and then add the employees as members to the corresponding workspaces.

Only administrators of an organization have permissions to manage members in the organization. By default, the creator of an organization is one of the administrators of the organization.

Members in an organization are classified into administrators and common members.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console**, or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Basics** tab.
4. Modify the organization information and click **Save**.

7.8.4. Leave an organization

Quick BI allows you to leave an organization.

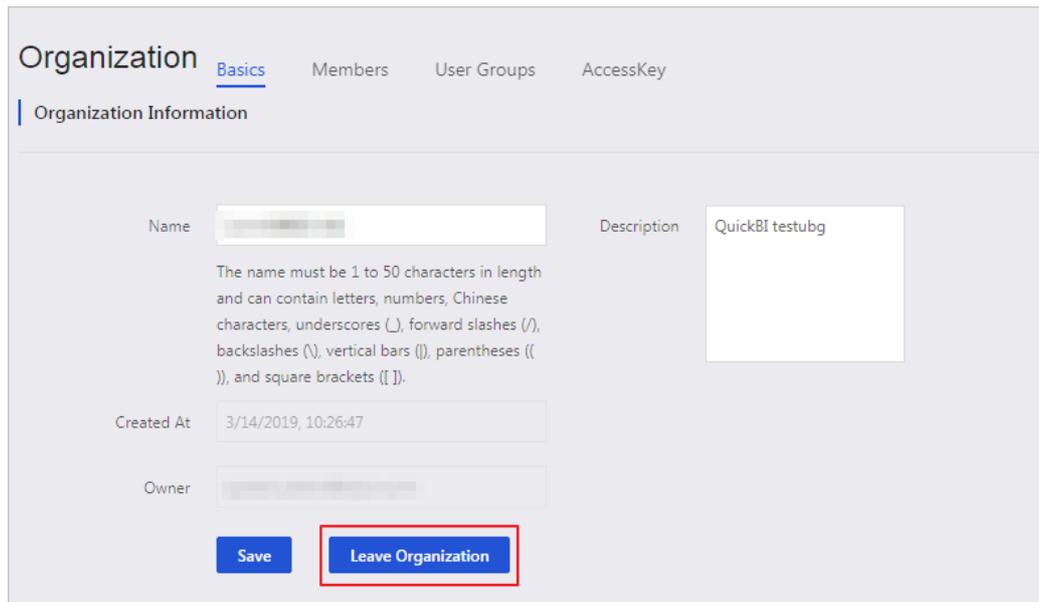
Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console**, or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Basics** tab and then click **Leave Organization**, as shown in [Leave an organization.](#)

Leave an organization



7.8.5. Add a member

You can add a member to an organization by **adding an Apsara Stack tenant account** or **adding a RAM user**.

Prerequisites

- The Quick BI service is purchased.
- The Apsara Stack tenant account is obtained.

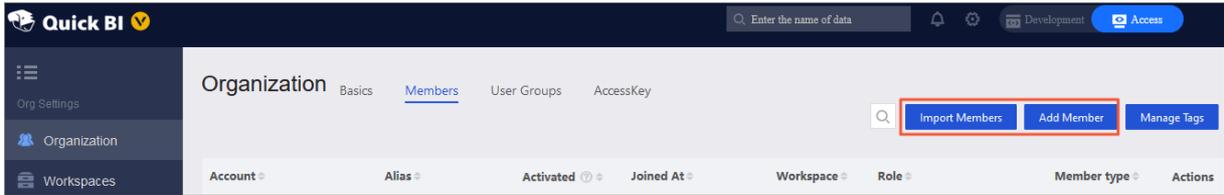
Each department created in Department Management of the Apsara Uni-manager Management Console has an Apsara Stack tenant account. The Apsara Stack tenant account of a user can be queried based on the department of the user.

- The RAM user information is obtained.

When you **add a RAM user** in the Quick BI console, you must know both the Apsara Stack tenant account and the RAM user. The RAM user is displayed on the User Management page.

Context

Quick BI allows you to add one member at a time or multiple members at the same time to an organization.



When you add one member at a time, you can add an Apsara Stack tenant account or a RAM user.

Add an Apsara Stack tenant account

Add Member

Tenant Account | RAM User

* Account Enter a valid Apsara Stack tenant account.
The account name cannot contain colons (:).

* Alias Enter a unique alias.
The alias must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

* Member type Developer Visitors
You can be added as a workspace member if and only if the organization member is a developer.

Role Set as Admin
 Set As Permission Administrator

Cancel OK

Add a RAM user

Add Member

Tenant Account RAM User

* Account Enter a valid Apsara Stack tenant account.
The account name cannot contain colons (:).

* RAM User Enter a valid RAM user.
The account name cannot contain colons (:).

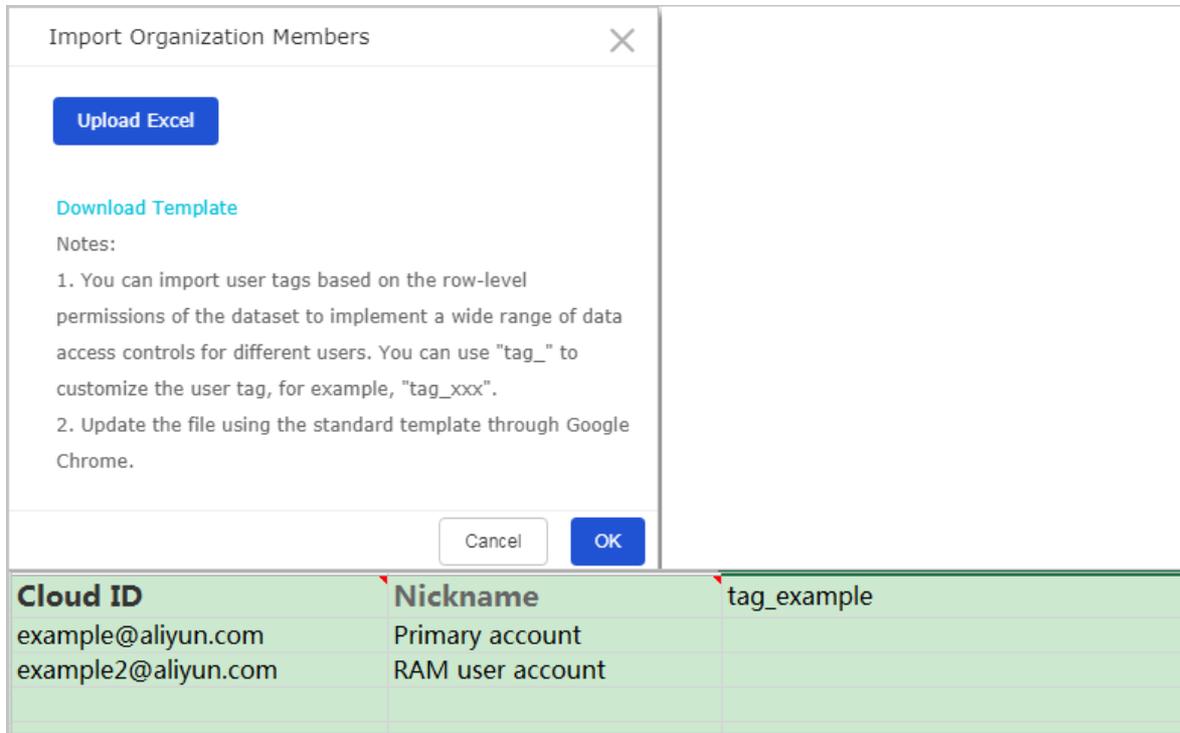
* Alias Enter a unique alias.
The alias must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

* Member type Developer Visitors
You can be added as a workspace member if and only if the organization member is a developer.

Role Set as Admin
 Set As Permission Administrator

Cancel OK

To add multiple members at the same time, you need to download a template and enter the Apsara Stack tenant accounts and aliases of the members you want to add into the template. Pay attention to the differences between the Apsara Stack tenant account and the RAM user. When you add a member by using a RAM user, the format of the user account is Apsara Stack tenant account:RAM user.



Add a member to an organization

Procedure

1. Log on to the Quick BI console.
2. On the homepage of the Quick BI console, or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Members** tab.
4. Click the **Add Member** tab.

Operation	Procedure
Tenant Account	<ol style="list-style-type: none"> In the Add Member dialog box, click the Tenant Account tab. Specify Account, Alias, and Member type, and select a role. Click OK.
RAM User	<ol style="list-style-type: none"> In the Add Member dialog box, click the RAM User tab. Specify Account, RAM User, Alias, and Member type, and select a role. Click OK. <p>Note If the user has been added to another organization, the system prompts an error.</p>

Add an Apsara Stack tenant account

Add Member ✕

Tenant Account **RAM User**

* **Account** Enter a valid Apsara Stack tenant account.
The account name cannot contain colons (:).

* **Alias** Enter a unique alias.
The alias must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

* **Member type** Developer Visitors
You can be added as a workspace member if and only if the organization member is a developer.

Role Set as Admin
 Set As Permission Administrator

Add a RAM user

Add Member

Tenant Account | **RAM User**

* **Account** Enter a valid Apsara Stack tenant account.
The account name cannot contain colons (:).

* **RAM User** Enter a valid RAM user.
The account name cannot contain colons (:).

* **Alias** Enter a unique alias.
The alias must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

* **Member type** Developer Visitors
You can be added as a workspace member if and only if the organization member is a developer.

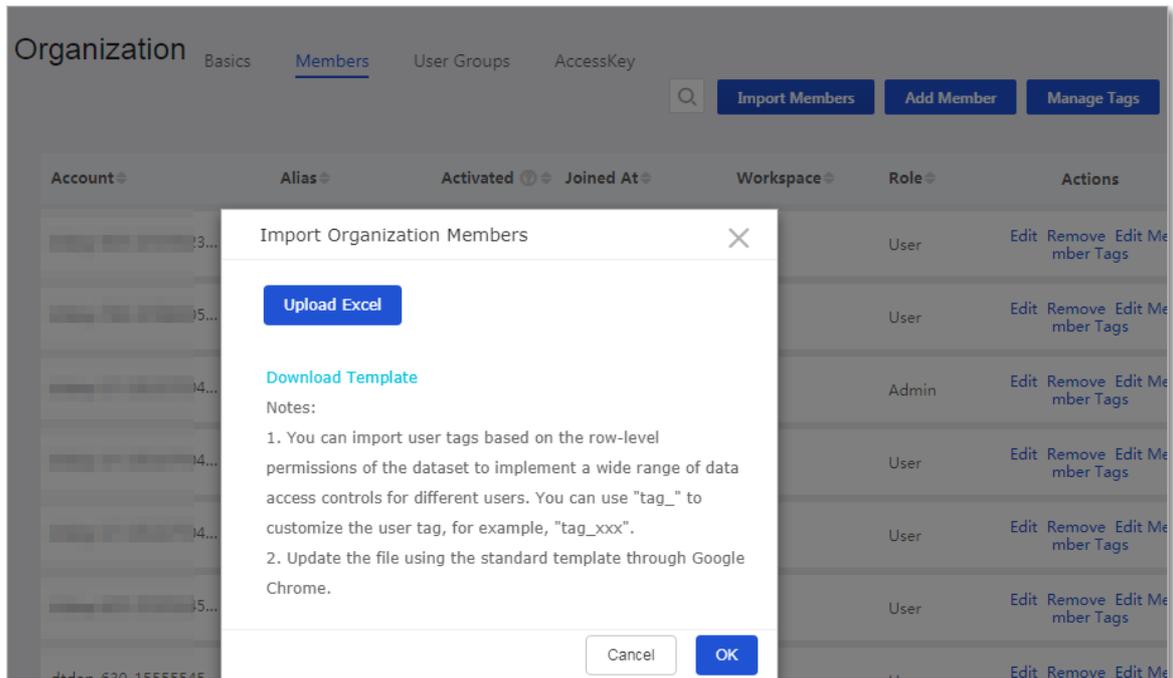
Role Set as Admin
 Set As Permission Administrator

Cancel OK

Add multiple members at a time

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Members** tab.
4. On the Members page, click **Import Members**.
5. In the Import Organization Members dialog box, click **Upload Excel** and select the local file that contains the member list.

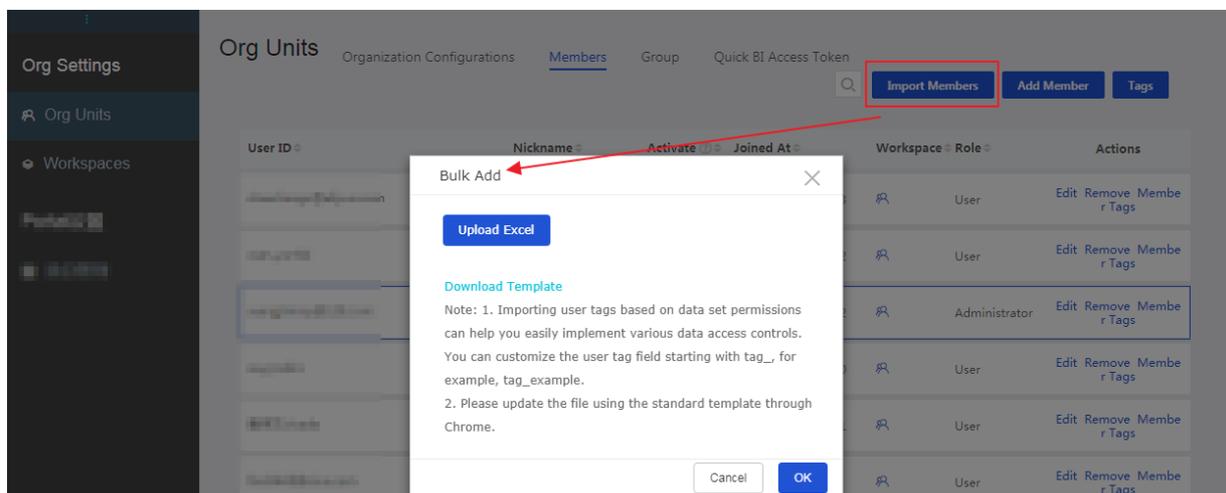


6. Click OK to add the members.

7.8.6. Manage member tags

This topic describes how to manage member tags, which are used to configure row-level permissions for datasets.

You can click the **Import Members** icon to add member tags.



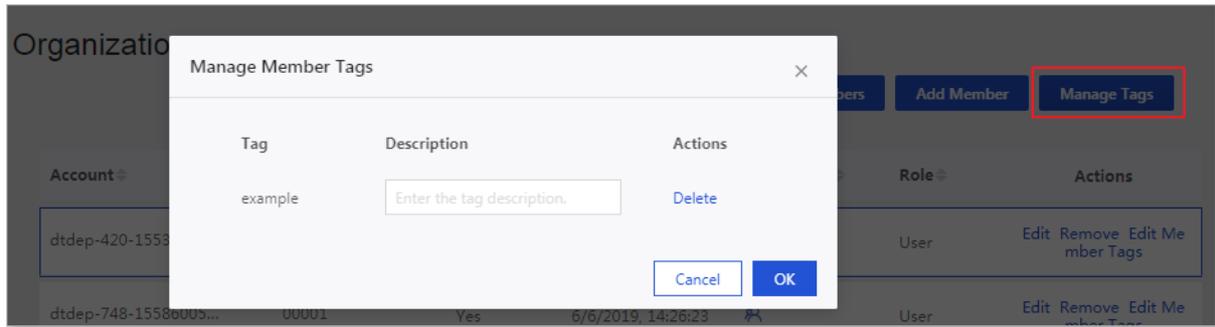
You can click **Download Template** to download the member template. The following figure shows member information.

Account	Nickname	tag_tagArea	tag_tagProvince
example1@aliyun.com	example1	East	Anhui
example2@aliyun.com	example2	East	Anhui

Note For members that do not require row-level permissions, you can set their tags to \$ALL_MEMBERS\$.

Manage member tags

After a tag is added, you can add, edit, or remove the tag by using tag management, as shown in the following figure.



7.8.7. Edit a member

You can set an alias and role for a member in an organization. This makes it easier for you to search for members in the organization.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Members** tab.
4. Find the member you want to edit and click the  icon in the Actions column.
5. Modify the information about the member.

Edit User Info

* Account

* Alias
The alias must be 1 to 50 characters in length and can contain letters, numbers, Chinese characters, underscores (_), forward slashes (/), backslashes (\), vertical bars (|), parentheses (()), and square brackets ([]).

* Member type Developer Visitors
You can be added as a workspace member if and only if the organization member is a developer.

Role Set as Admin
 Set As Permission Administrator

Cancel OK

6. Click OK to save the changes.

7.8.8. Remove a member

This topic describes how to remove a member from an organization.

Prerequisites

The Quick BI service is purchased.

Context

Only administrators of an organization have the permissions to remove members from the organization. Before you remove a member that has been added to a workspace, you must remove the member from the workspace. Otherwise, you cannot remove the member from the organization.

Note The removal operation is irreversible. If you need to restore a member that you previously removed, you must add the member to the organization again. Remove a member with caution.

Procedure

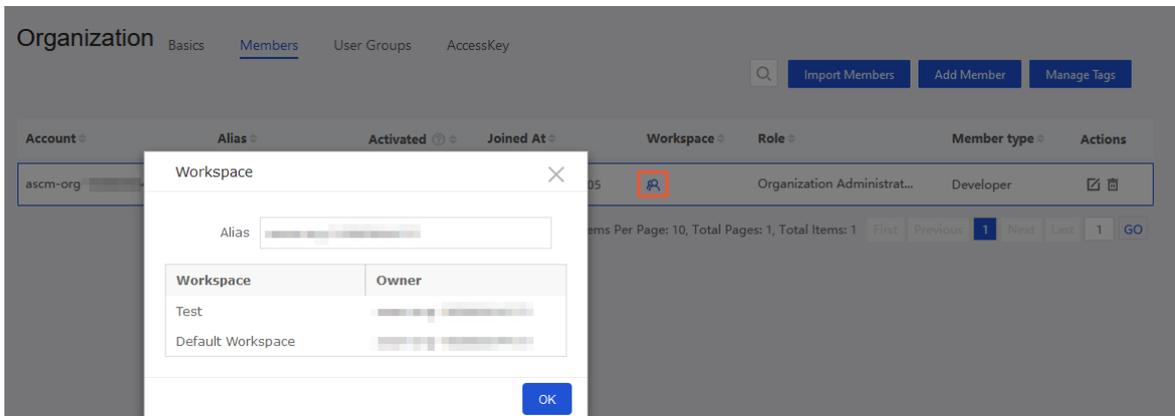
1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the  icon.
3. On the Organization page, click the **Members** tab.
4. Select a member and click the  icon in the Actions column.
5. Click **OK** to remove the member.

7.8.9. Query the workspace to which a user belongs

You can query the workspace to which a user belongs.

Procedure

1. Log on to the Quick BI console.
2. On the homepage of the Quick BI console or in a workspace, click the  icon.
3. On the Organization page, click the **Members** tab.
4. Find the organization member that you want to query and click the  icon in the **Workspace** column to view the workspace to which the user belongs.

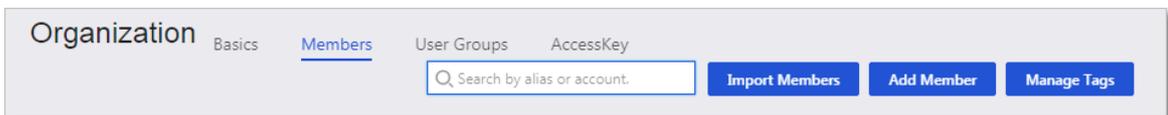


7.8.10. Search for a member of an organization

You can search for a specific member in an organization by its alias or Alibaba Cloud account.

Procedure

1. Log on to the Quick BI console.
2. On the homepage of the Quick BI console or in a workspace, click the **Settings** icon.
3. On the Organization page, click the **Members** tab.
4. Enter the alias or Apsara Stack tenant account of the member you want to search for in the search box, and click the **Search** icon.



7.8.11. Create a user group

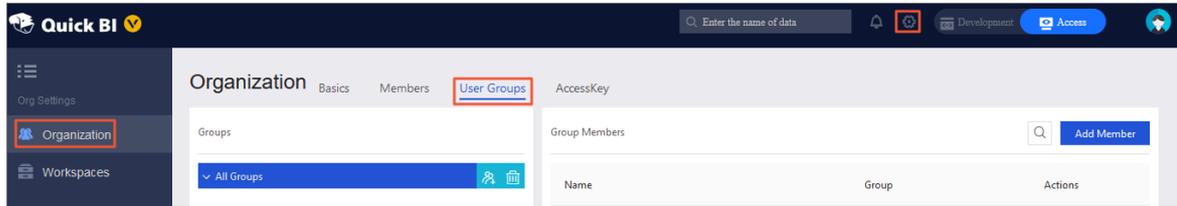
To facilitate setting permissions for different organization members, you can create different user groups based on the departments (such as the sales, operations, and technology departments) or the roles (such as analysts, project managers, and project directors) of the members. The same member can belong to multiple user groups.

Prerequisites

The Quick BI service is purchased.

Procedure

1. Log on to the Quick BI console.
2. On the homepage of the Quick BI console or in a workspace, click the  icon.
3. On the Organization page, click the User Groups tab.



4. On the User Groups page, click the  icon to customize the Name of a child group, and then click OK.

Add Child Group ×

Parent Group Groups

Name

7.8.12. Manage theme templates

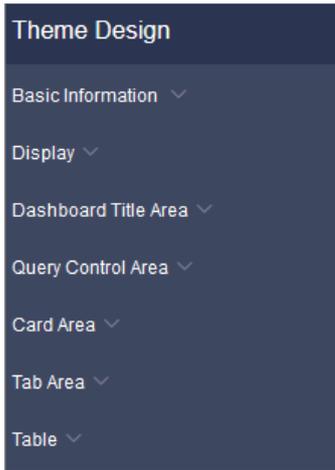
This topic describes how to add custom theme templates or manage theme templates for a dashboard.

Add a theme template

1. Log on to the Quick BI console.
2. In the target workspace, click the  icon.
3. In the left-side navigation pane, click Theme template.
4. On the Theme Template Management page, click Add Theme.

 **Note** A maximum of five custom theme templates are supported.

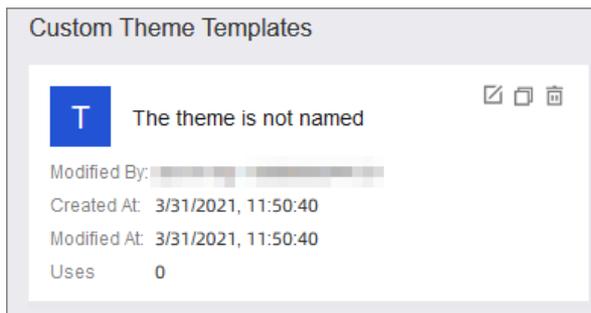
5. On the Theme Design page, configure **Basic Information**, **Display**, **Dashboard Title Area**, **Query Control Area**, **Card Area**, **Tab Area**, and **Table**. For more information, see [Theme design](#).



6. Click **Save**. You can refresh the **Theme Template Management** page to view the created theme template.

Manage custom theme templates

On the **Theme Template Management** page, find the required theme template and perform the following operations.



- Click the  icon to edit a theme template.
- Click the  icon to copy a theme template.
- Click the  icon to delete a theme template.

7.8.13. Workspaces

7.8.13.1. Overview

A workspace is managed by its administrators. The role of a workspace administrator is assigned to members by the organization administrator that creates the workspace. A workspace administrator can specify other members in the workspace as administrators of the workspace.

A workspace administrator can:

- Create a workspace
- Modify a workspace
- Set a default workspace

7.8.13.2. What is a workspace?

A workspace enables members in the same organization to collaborate on tasks. In a workspace, each member is assigned a role to perform operations such as creating and modifying data sources, datasets, workbooks, dashboards, and BI portals. Data objects are stored in their workspaces. Each workspace has its own data objects.

A workspace has the following properties:

- Name
- Description
- Function permissions: Data objects can be shared and made public by default.
- Preference settings:
 - Use physical field names as dimension and measurement names.
 - Use field annotations as dimension and measurement names.

Datasets created in this workspace are named based on the new settings. Existing datasets are not affected.

Members of a workspace can be assigned the same role or different roles. The roles in a workspace include:

- Administrator
- Developer
- Analyst
- Viewer

Mapping between roles and permissions

Permissions of a role cannot be changed. If you want to grant specific permissions to a member, you only need to assign the appropriate role to the member. The following tables list the permissions of each role.

- Supported operations on data objects Supported operations on data objects

Permission	Developer	Analyst	Viewer
Dataset	Yes	No	No
Supported operations on workbooks	Yes	Yes	Yes
Dashboards	Yes	Yes	Yes
BI Portals	Yes	Yes	Yes

- Supported operations on data Supported operations on data

Permission	Developer	Analyst	Viewer
Create a data source	Yes	No	No
Modify a data source	Developers can modify only data sources created by themselves.	No	No
Delete a data source	Developers can only delete data sources created by themselves.	No	No
Use a data source	Yes	No	No

Permission	Developer	Analyst	Viewer
Create a dataset	Yes	No	No
Modify a dataset	Developers can modify only datasets created by themselves.	No	No
Delete a dataset	Developers can only delete datasets created by themselves.	No	No
Use a dataset	Yes	Yes	No

• Supported operations on workbooks Supported operations on workbooks

Permission	Developer	Analyst	Viewer
Create a workbook	Yes	Yes	No
Modify a workbook	Developers can modify only workbooks created by themselves.	Analysts can modify only workbooks created by themselves.	No
Delete a workbook	Developers can only delete workbooks created by themselves.	Analysts can only delete workbooks created by themselves.	No
Preview a workbook	Yes	Yes	Yes
Share a workbook	Developers can only share workbooks created by themselves.	Analysts can only share workbooks created by themselves.	No
Reference a workbook	Yes	Yes	No

• Supported operations on dashboards Supported operations on dashboards

Permission	Developer	Analyst	Viewer
Create a dashboard	Yes	Yes	No
Modify a dashboard	Developers can modify only dashboards created by themselves.	Analysts can modify only dashboards created by themselves.	No
Delete a dashboard	Developers can only delete dashboards created by themselves.	Analysts can only delete dashboards created by themselves.	No
View a dashboard	Yes	Yes	Yes
Share a dashboard	Developers can only share dashboards created by themselves.	Analysts can only share dashboards created by themselves.	No
Reference a dashboard	Yes	Yes	No

Permission	Developer	Analyst	Viewer
Publish a dashboard	Developers can only publish dashboards created by themselves.	Analysts can only publish dashboards created by themselves.	No

• Supported operations on BI portals Supported operations on BI portals

Permission	Developer	Analyst	Viewer
Create a BI portal	Yes	Yes	No
Modify a BI portal	Developers can modify only BI portals created by themselves.	Analysts can modify only BI portals created by themselves.	No
Delete a BI portal	Developers can only delete BI portals created by themselves.	Analysts can only delete BI portals created by themselves.	No
View a BI portal	Yes	Yes	Yes
Share a BI portal	Developers can only share BI portals created by themselves.	Analysts can only share BI portals created by themselves.	No

7.8.13.3. Differences between a personal workspace and a group workspace

Workspaces of a user are classified into a personal workspace and group workspaces. The personal workspace is automatically created when the user first logs on to the Quick BI console. A group workspace is created for group members to collaborate. Differences between the two types of workspaces are as follows:

- A personal workspace is automatically created when you log on to the Quick BI console for the first time. A group workspace needs to be manually created by an organization administrator.
- You cannot create or delete a personal workspace.
- You cannot add other users to your personal workspace, nor can you collaboratively edit or transfer it.
- A group workspace can be transferred to any member in the group workspace and shared with all members in the organization. A personal workspace can be shared with Apsara Stack tenant accounts.

7.8.14. Create a workspace

This topic describes how to create a workspace.

Prerequisites

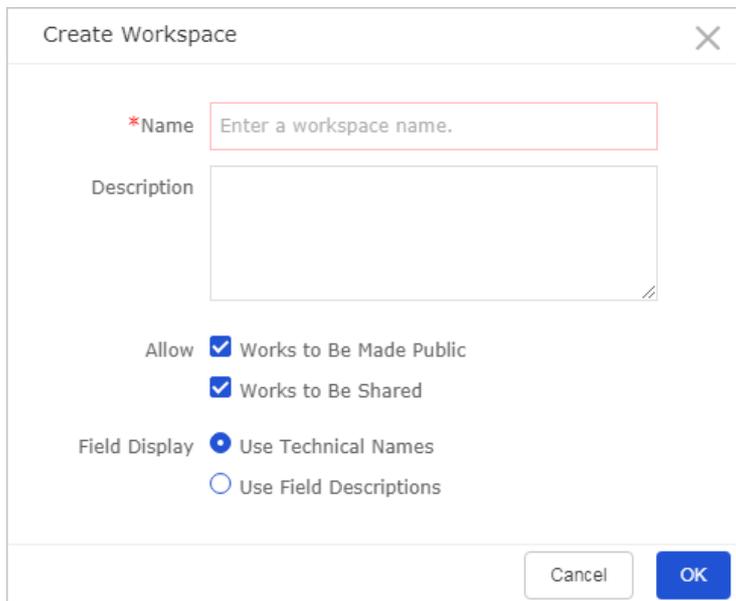
The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. On the Organization page that appears, click **Workspaces** in the left-side navigation pane. Then, click

Create Workspace.

4. In the Create Workspace dialog box that appears, enter a name for the workspace.



The screenshot shows a 'Create Workspace' dialog box. It features a title bar with a close button (X). The main content area includes a text input field for 'Name' with a red border and placeholder text 'Enter a workspace name.'. Below this is a larger text area for 'Description'. There are two sections of checkboxes: 'Allow' with 'Works to Be Made Public' and 'Works to Be Shared' both checked; and 'Field Display' with 'Use Technical Names' selected and 'Use Field Descriptions' unselected. At the bottom right are 'Cancel' and 'OK' buttons.

5. Click **OK** to create the workspace.

7.8.15. Modify information about a workspace

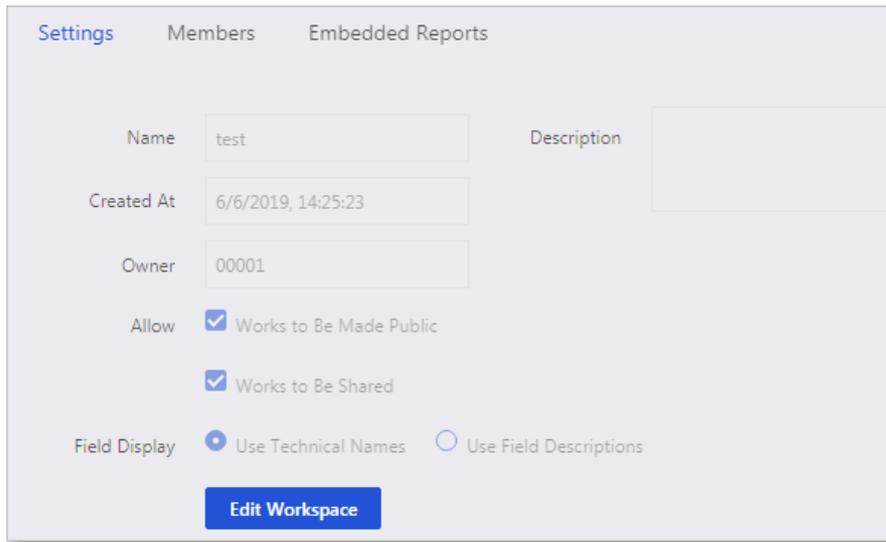
For a personal workspace, only its owner can modify the workspace information. For a group workspace, only its administrators can modify the workspace information.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. In the left-side navigation pane of the Organization page that appears, click **Workspaces**.
4. Click the **Settings** tab.
5. Click **Edit Workspace** and modify the information about the workspace.



6. Click OK.

7.8.16. Leave a workspace

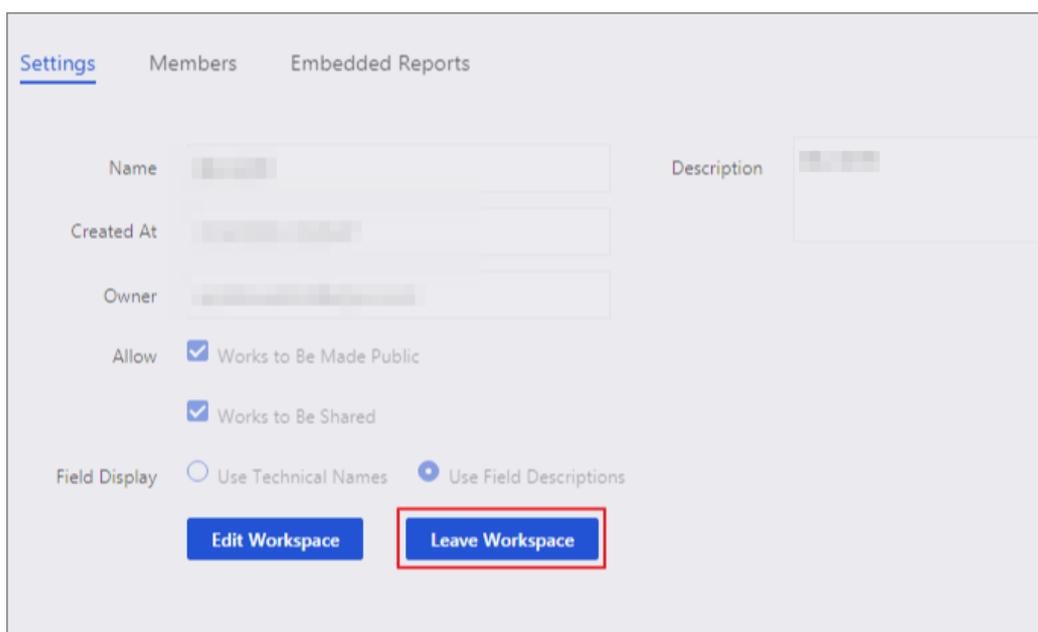
This topic describes how to leave a workspace.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. In the left-side navigation pane of the Organization page that appears, click **Workspaces**.
4. Click the workspace that you want to leave and click the **Settings** tab.
5. Click **Leave Workspace** to leave the current workspace.



7.8.17. Transfer a workspace to another member

This topic describes how to transfer a workspace to another member.

Prerequisites

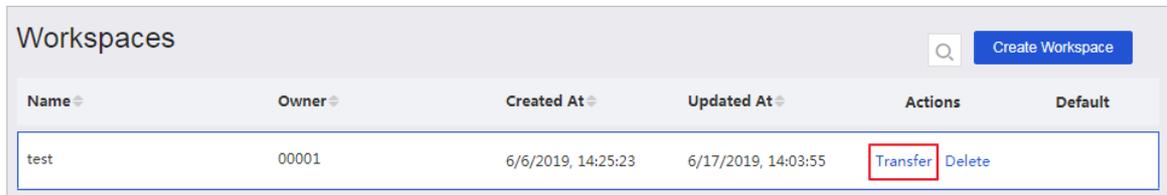
The Quick BI service is purchased.

Context

If you need to remove the owner of a workspace from the organization, you can transfer the workspace to another member. The new owner can be any member in the organization.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. In the left-side navigation pane of the Organization page that appears, click **Workspaces**.
4. Find the workspace that you want to transfer and click **Transfer** in the Actions column.



5. Enter the alias of the new owner and click **OK**.

7.8.18. Delete a workspace

This topic describes how to delete a workspace.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. In the left-side navigation pane of the Organization page that appears, click **Workspaces**.
4. On the Workspaces page, find the workspace that you want to delete and click **Delete** in the Actions column.



7.8.19. Add a member to a workspace

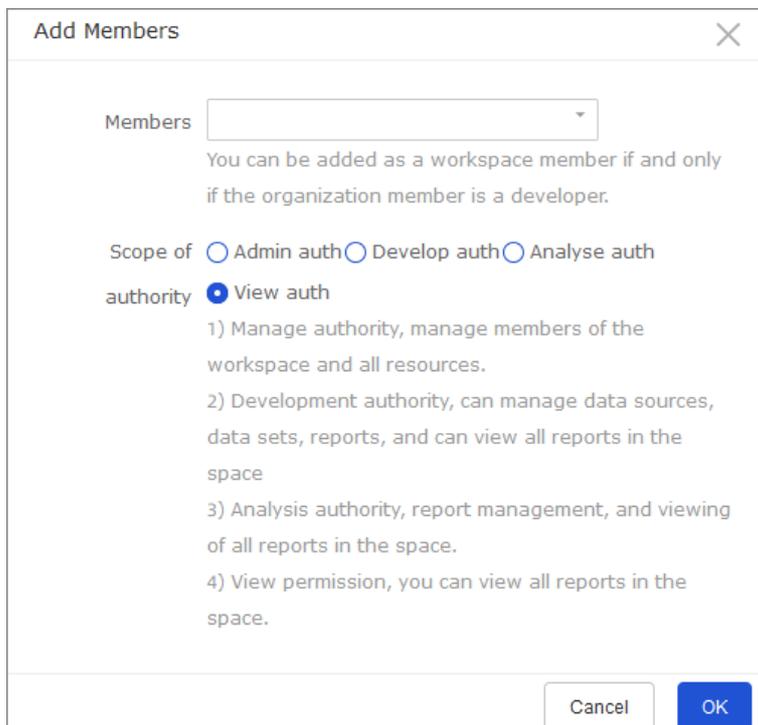
This topic describes how to add a member to a workspace.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the  icon.
3. Click **Workspaces** in the left-side navigation pane.
4. On the **Workspaces** page, click the **Members** tab.
5. Click **Add Members**.
6. Enter the account of the member that you want to add and assign a role to the account.



Add Members [X]

Members

You can be added as a workspace member if and only if the organization member is a developer.

Scope of authority

Admin auth Develop auth Analyse auth View auth

1) Manage authority, manage members of the workspace and all resources.

2) Development authority, can manage data sources, data sets, reports, and can view all reports in the space

3) Analysis authority, report management, and viewing of all reports in the space.

4) View permission, you can view all reports in the space.

Cancel OK

7.8.20. Edit settings of a workspace member

This topic describes how to edit the settings of a workspace member.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. In the left-side navigation pane of the Organization page that appears, click **Workspaces**. On the **Workspaces** page that appears, click the **Members** tab.
4. Find the member whose settings you want to edit and click **Edit** in the Actions column.
5. Change the settings of the member and click **OK**.

7.8.21. Search for a member in a workspace

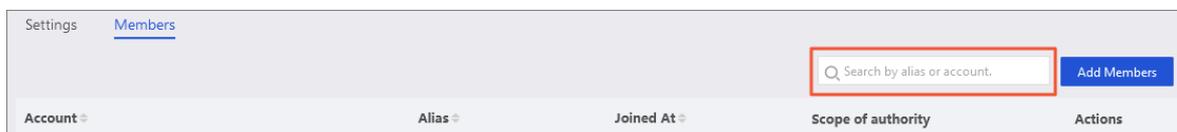
You can search for a member on the **Members** tab.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. Click **Workspaces**.
4. On the **Workspaces** page, click the **Members** tab.
5. Enter an alias or account in the search box.
6. Click the  icon to search for a member.



7.8.22. Delete a member from a workspace

This topic describes how to delete a member from a workspace.

Prerequisites

The Quick BI service is purchased.

Procedure

1. [Log on to the Quick BI console.](#)
2. On the homepage of the **Quick BI console** or in a workspace, click the **Settings** icon.
3. Click **Workspaces** in the left-side navigation pane.
4. On the **Workspaces** page, click the **Members** tab.
5. Find the member that you want to delete and click **Delete** in the **Actions** column.
6. In the **Remove Member** dialog box, select a member as the new owner from the **Specify a new owner** drop-down list.
7. Click **OK**.

7.9. Permissions

7.9.1. Overview of permissions

You can manage Quick BI permissions at the data object and row levels.

Data object-level permission control

The data objects you can manage in Quick BI include data sources, datasets, dashboards, workbooks, and BI portals. You can manage them in personal workspaces or group workspaces. For information about differences between personal workspaces and group workspaces, see [Data objects](#).

Row-level permission control

You do not need to configure row-level permissions for all fields in a dataset. Configure row-level permissions for specific fields based on your business needs.

 **Note** Only datasets in group workspaces support row-level permission control.

7.9.2. Manage data objects

This topic describes how to manage data objects. Data objects include data sources, datasets, dashboards, workbooks, and BI portals.

The following table lists differences between data object management in a personal workspace and that in a workspace, also referred to as a group workspace.

Item	Workspace (Group workspace)	Personal workspace
Permissions	<p>Data objects can be shared or made public.</p> <p> Note Data sources and datasets cannot be shared or made public.</p>	<p>Only the owner of a personal workspace can perform operations on data objects.</p>
Share data objects	<p>Workbooks, dashboards, and BI portals can be shared. Shared data objects are read-only for other Apsara Stack tenant accounts and RAM users. Other Apsara Stack tenant accounts and RAM users cannot modify, delete or save data objects.</p> <ul style="list-style-type: none"> Only the data object owner and the administrators of the workspace have the permissions to share the data object. If the Works to Be Shared check box is cleared in workspace settings, data objects in the workspace cannot be shared. Data objects can only be shared with Apsara Stack tenant accounts and RAM users of the same organization. <p>Members can view the data objects in the workspace to which the members belong. Data objects can be shared with users in different workspaces within an organization. Authorized users can view the shared data objects in their own personal workspaces.</p>	<p>Workbooks, dashboards, and BI portals can be shared. Shared data objects are read-only for other Apsara Stack tenant accounts and RAM users. Other Apsara Stack tenant accounts and RAM users cannot modify, delete or save data objects.</p> <ul style="list-style-type: none"> Only data object owners have the permissions to share the data objects. Data objects can only be shared with users of Apsara Stack Quick BI. <p>Authorized users can view shared data objects in their own personal workspaces.</p>
Make data objects public	<p>Everyone can access data objects that have been made public by using URLs. We recommend that you do not make the data objects that contain private business data public.</p>	<p>Everyone can access data objects that have been made public by using URLs. We recommend that you do not make the data objects that contain private business data public.</p>

7.9.3. Manage row-level permissions

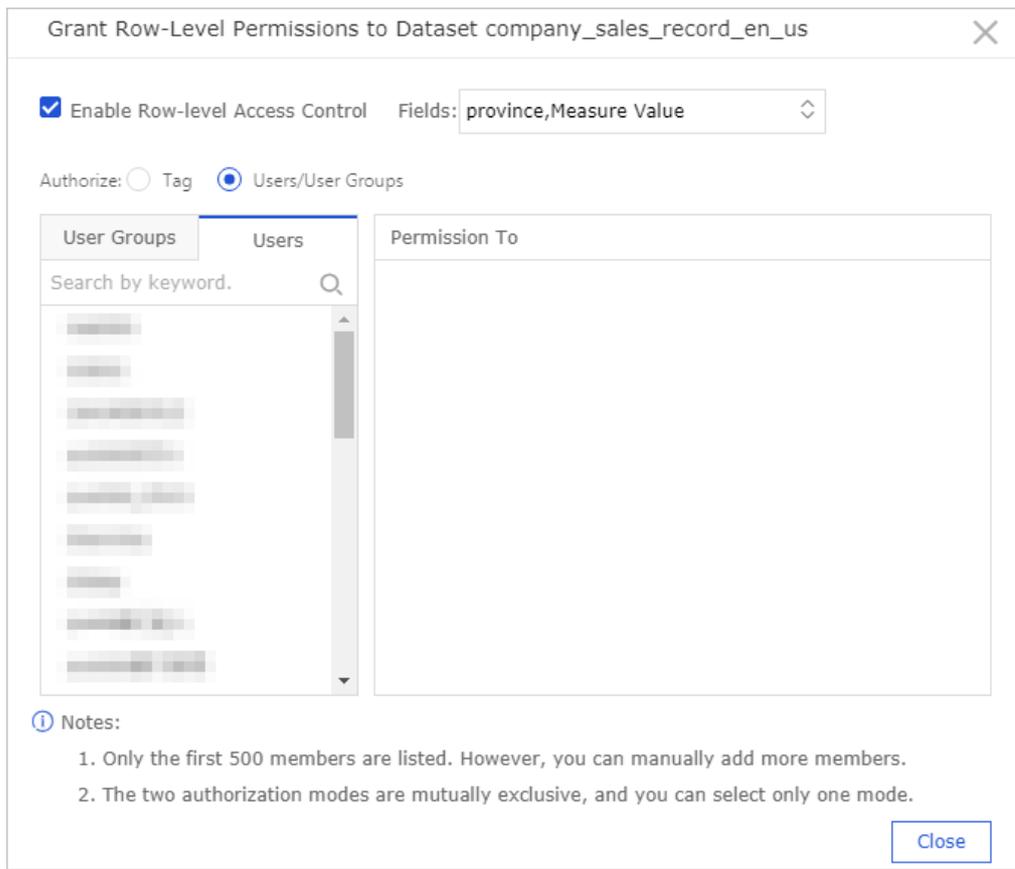
Row-level permissions are managed based on datasets. Quick BI supports the following authorization modes: **user/user group-based authorization** and **tag-based authorization**. **User/user group-based authorization** is ideal for organizations with a small number of members. **Tag-based authorization** is ideal for organizations with a large number of members.

User/user group-based authorization

1. [Log on to the Quick BI console](#)
2. Select a workspace. For information about how to create a workspace, see [Create a workspace](#).
3. In the left-side navigation pane of the Workspace page, click **Datasets**.

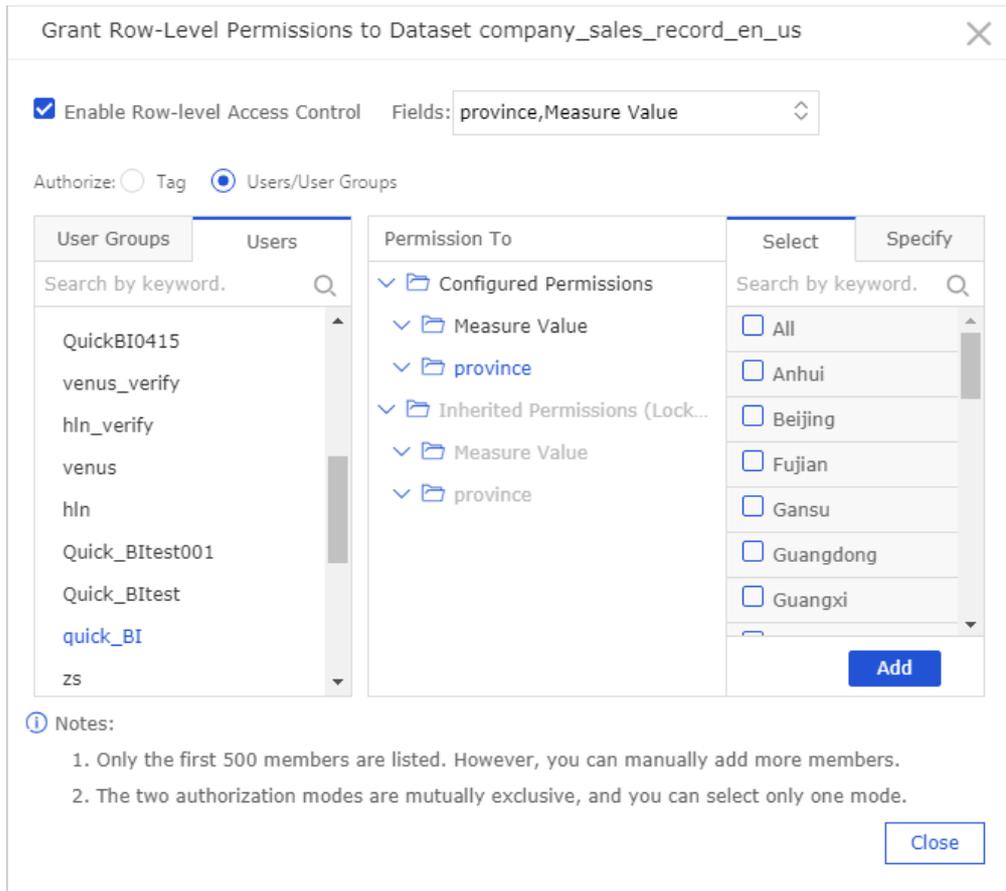
Note Datasets in a personal workspace do not support row-level permission control.

4. Find the target dataset. Click the **More** icon in the Actions column or right-click the dataset.
5. Select **Grant Row-Level Permissions**.
6. On the Grant Row-Level Permissions page, select **Enable Row-level Access Control** and select **Users/User Groups** for Authorize.
7. Click the drop-down arrow of Fields. Select the fields that the authorization is based on, such as province and Measure Value, as shown in the following figure.



The values of the **Measure Value** field are the measures in the dataset. By granting row-level permissions based on the Measure Value field, you can specify the measures available to different users.

8. In the **Permission To** section, click the **province** field. A list of provinces appear.
9. Select a member and choose values of the **province** field to grant permissions to the member, as shown in the following figure.



In this example, the member can view the data of Shanghai and Yunnan.

Note If you grant permissions based on a field of a dataset, you must specify whether all members of the workspace have permissions to access the dataset. Otherwise, when other users attempt to access reports created based on the dataset, the system denies the access requests by default.

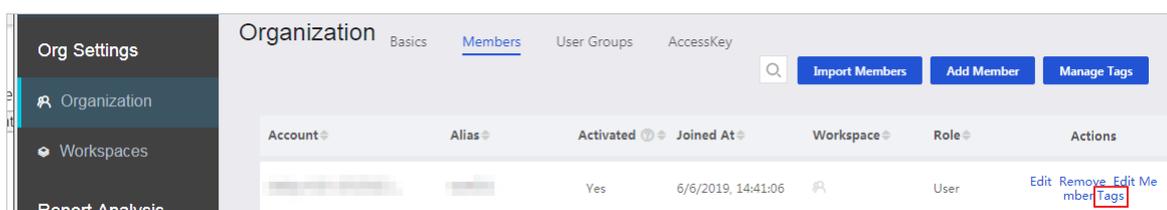
10. Click **Add** to complete the authorization.

Tag-based authentication

Scenario: Users can only access the data rows in the company_sales_record dataset with shipping_type set to truck and air.

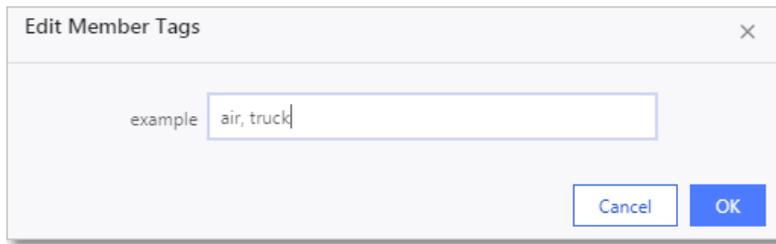
Set member tags

1. In the target workspace, click the **Settings** icon.
2. On the **Settings** page, find the member you want to authorize and click **Edit Member Tags** in the Actions column.



3. In the **Edit Member Tags** dialog box that appears, set the value of the example tag to air, truck and

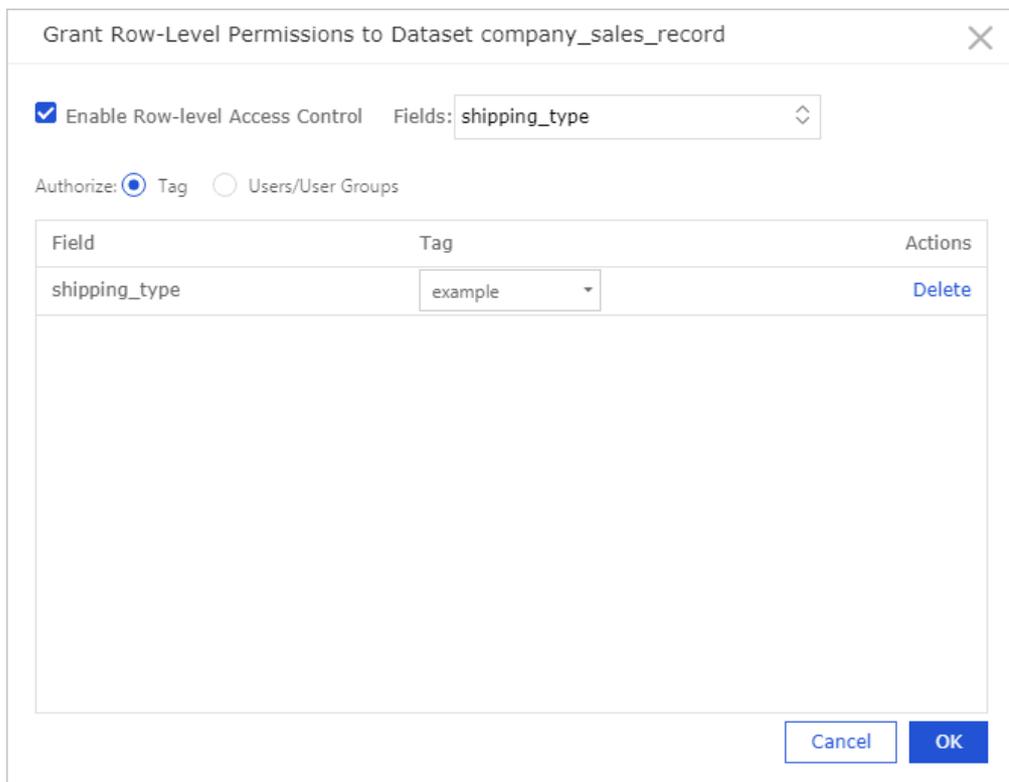
click **OK**.



After you set the member tag, you must specify the tag in the Grant Row-Level Permissions dialog box.

Set tag-based authorization

1. Find the `company_sales_record` dataset. Click the **More** icon in the Actions column or right-click the dataset.
2. Select **Grant Row-level Permissions**.
3. On the Grant Row-Level Permissions page, select **Enable Row-level Access Control** and select **Tag** for Authorize.
4. From the **Fields** drop-down list, select `shipping_type`. Select `example` in the **Tag** column, and click **OK** to complete the settings.



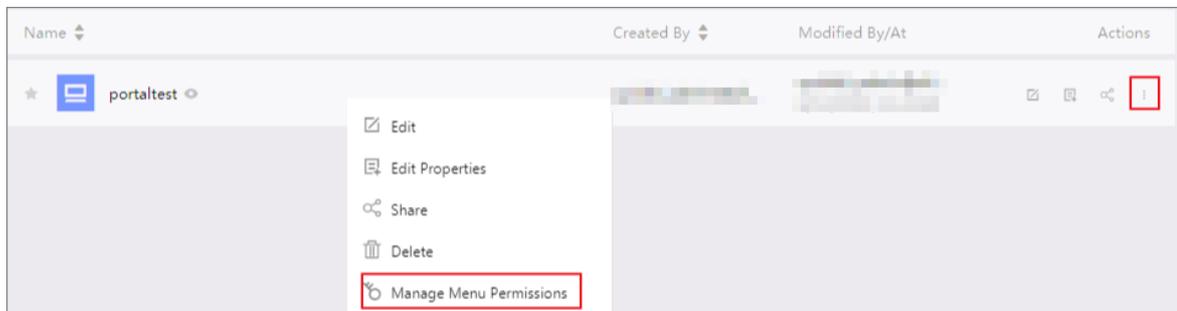
After tag authorization is complete, the user can only access data rows with `shipping_type` set to air or truck.

7.9.4. Configure menu permissions for a BI portal

Workspace administrators can configure permissions on viewing BI portal menus in a workspace, also referred to as a group workspace.

The permissions can be granted to user groups or individual users by performing the following steps:

1. [Log on to the Quick BI console.](#)
2. Select a workspace. For information about how to create a workspace, see [Create a workspace.](#)
3. In the left-side navigation pane of the Workspace page, click **BI Portals**.
4. On the BI Portals page, find the BI portal for which you want to configure the menu permissions, click the **More** icon in the Actions column, or right-click the BI portal, and select **Manage Menu Permissions**.



5. In the **Manage Menu Permissions** dialog box, select the menu for which you want to manage the permissions, specify whether the menu is available only to authorized users, and select the users or user groups that you want to authorize.

Note The values of **Available Only to Authorized Users** are described as follows:

- o Yes: Only authorized users and user groups can access this menu.
- o No: All users and user groups can access this menu.

6. Click **Close** to complete configuring the menu permissions.

7.9.5. Share a data object in a personal workspace

Only the owner of a data object has the permissions to share the data object.

Prerequisites

The Quick BI service is purchased.

Context

In a personal workspace, you can share **workbooks**, **dashboards**, and **BI portals**. Shared data objects are read-only for other Apsara Stack tenant accounts and RAM users. Other Apsara Stack tenant accounts and RAM users do not have the permissions to modify, delete, or save the data objects.

Members with whom a data object is shared can log on to the Quick BI console and view the data object on the **My Items** page.

This topic uses a dashboard as an example to describe how to share data objects in a personal workspace.

Procedure

1. [Log on to the Quick BI console.](#)
2. In the left-side navigation pane of the Workspace page, click **Dashboards**.
3. On the Dashboards page, select the dashboard that you want to share with others and click the **Share** icon in the Actions column.
4. Enter the account of the user that you want to share the dashboard with, and specify an expiration date.
5. Click **Save** to share the dashboard.

7.9.6. Share a data object in a workspace

In a workspace, you can share **workbooks**, **dashboards**, and **BI portals**. Shared data objects are read-only for other Apsara Stack tenant accounts and RAM users. Other Apsara Stack tenant accounts and RAM users do not have the permissions to modify, delete, or save the data objects.

Prerequisites

The Quick BI service is purchased.

Context

Only the owner of a data object or administrators of the workspace have the permissions to share the data object. Data objects can only be shared with Apsara Stack tenant accounts and RAM users within the same organization.

If the Works to Be Shared check box is cleared for a workspace, the data objects in the workspace cannot be shared.

This topic takes a dashboard as an example to describe how to share data objects in a workspace.

Procedure

1. [Log on to the Quick BI console.](#)
2. Select a workspace.
3. In the left-side navigation pane of the Workspace page, click **Dashboards**.
4. On the Dashboards page, select the dashboard you want to share with others and click the **Share** icon in the Actions column.
5. Enter the alias or account of the user that you want to share the dashboard with, and specify an expiration date.
6. Click **Save** to share the dashboard.

7.9.7. Publish data objects that are stored in a personal workspace

You can publish data objects that are stored in a personal workspace. All Internet users can visit the URLs that point to published data objects. We recommend that you do not publish data objects that include sensitive business data.

Prerequisites

You have purchased Quick BI.

Context

In a personal workspace, you can publish **dashboards** and **workbooks**.

This topic takes a dashboard as an example to describe how to publish data objects in a personal workspace.

Procedure

1. [Log on to the Quick BI console.](#)
2. Select a workspace.
3. Click **Dashboards** to go to the Dashboards page.

4. Select the target dashboard and click the **More** icon and select **Make Public**.
5. Specify an expiration date and click **Make Public**.

A URL is generated and appears in the **Make Public** dialog box. You can copy and paste the URL into the address bar of your browser, and then access the dashboard by using the URL.

7.9.8. Make a data object in a workspace public

This topic describes how to make a data object in a workspace, also referred to as a group workspace, public. All Internet users can visit public data objects by using the provided URLs. We recommend that you do not publish data objects that contain sensitive business data.

Prerequisites

The Quick BI service is purchased.

Context

In a workspace, you can make **dashboards** and **workbooks** public.

This topic uses a dashboard as an example to describe how to make data objects in a workspace public.

Procedure

1. [Log on to the Quick BI console.](#)
2. Select a workspace.
3. In the left-side navigation pane of the Workspace page, click **Dashboards**.
4. On the Dashboards page, select the dashboard that you want to make public, click the **More** icon in the Actions column, and select **Make Public**.
5. In the **Make Public** dialog box, specify an expiration date and click **Make Public**.

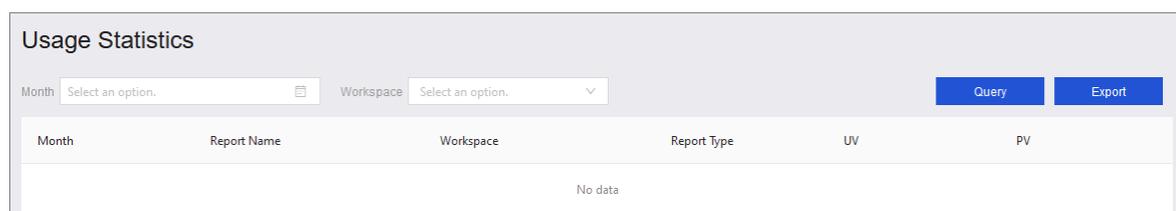
A URL is generated and appears in the **Make Public** dialog box. You can copy and paste the URL into the address bar of your browser to access the dashboard.

7.10. Report statistics

7.10.1. Usage statistics

Usage statistics allow you to track the Unique Visitor (UV) and Page View (PV) values of a specific report in a specific month.

1. [Log on to the Quick BI console.](#)
2. On the homepage of the Quick BI, click the  icon in the top navigation bar.
3. In the left-side navigation pane, click **Usage Statistics**.
4. On the Usage Statistics page, select a month and workspace and click **Query**.

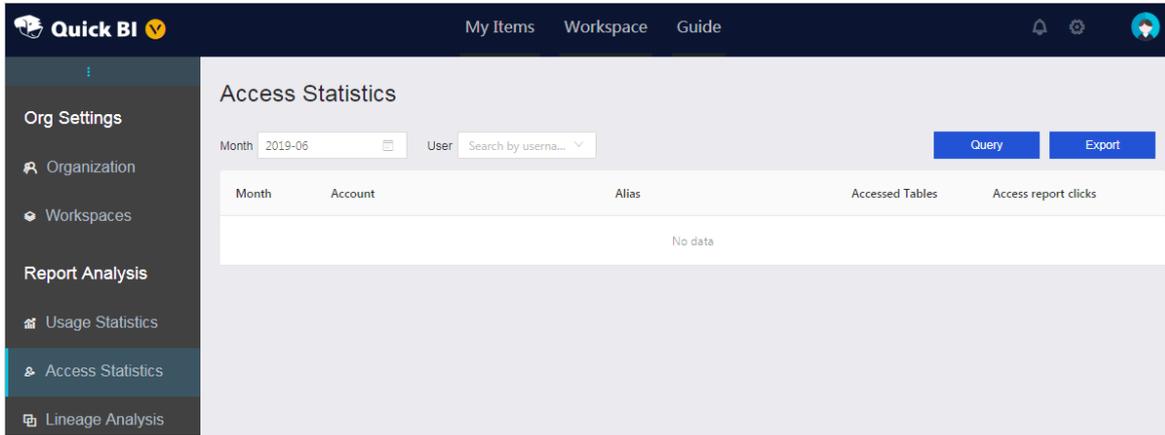


5. Click **Export** to export the statistics data to a local device as an Excel file.

7.10.2. Access statistics

Access statistics allow you to track the number of reports that you have accessed in a specific month and the number of clicks on a specific report.

1. [Log on to the Quick BI console.](#)
2. On the homepage of the Quick BI, click the  icon in the top navigation bar.
3. In the left-side navigation pane, click **Access Statistics**.
4. On the Access Statistics page, select a month, enter the member alias, and then click **Query**.

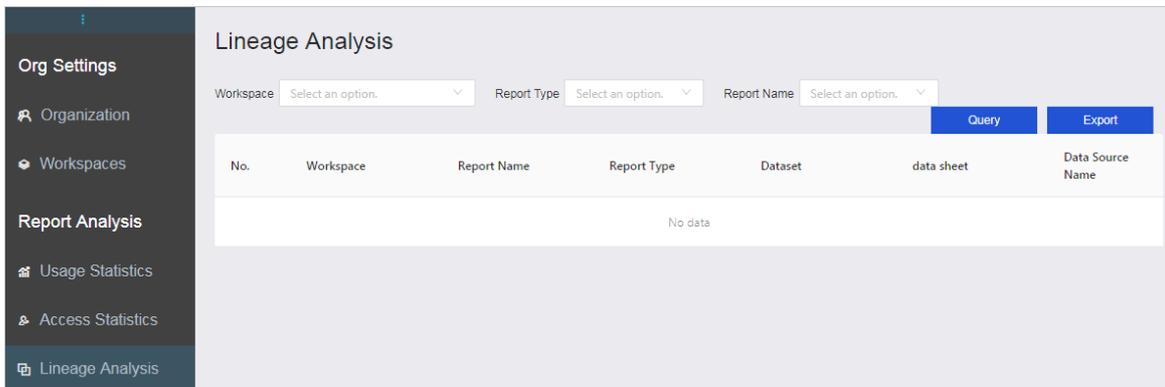


5. Click **Export** to export the statistics data as an Excel file to a local device.

7.10.3. Lineage analysis

Lineage analysis allows you to query information about a specific report, such as the workspace, report type, dataset, chart, and data source name.

1. On the homepage of the Quick BI console, click the **Settings** icon next to the profile picture.
2. In the left-side navigation pane, click **Lineage Analysis**.
3. On the Lineage Analysis page, select the workspace, report type, and report name of the report you want to query, and click **Query**.



4. Click **Export** to export the analysis results as an Excel file to a local device.

7.11. Open service

7.11.1. Overview of the open service

The open service allows third parties to access Quick BI reports by embedding reports.

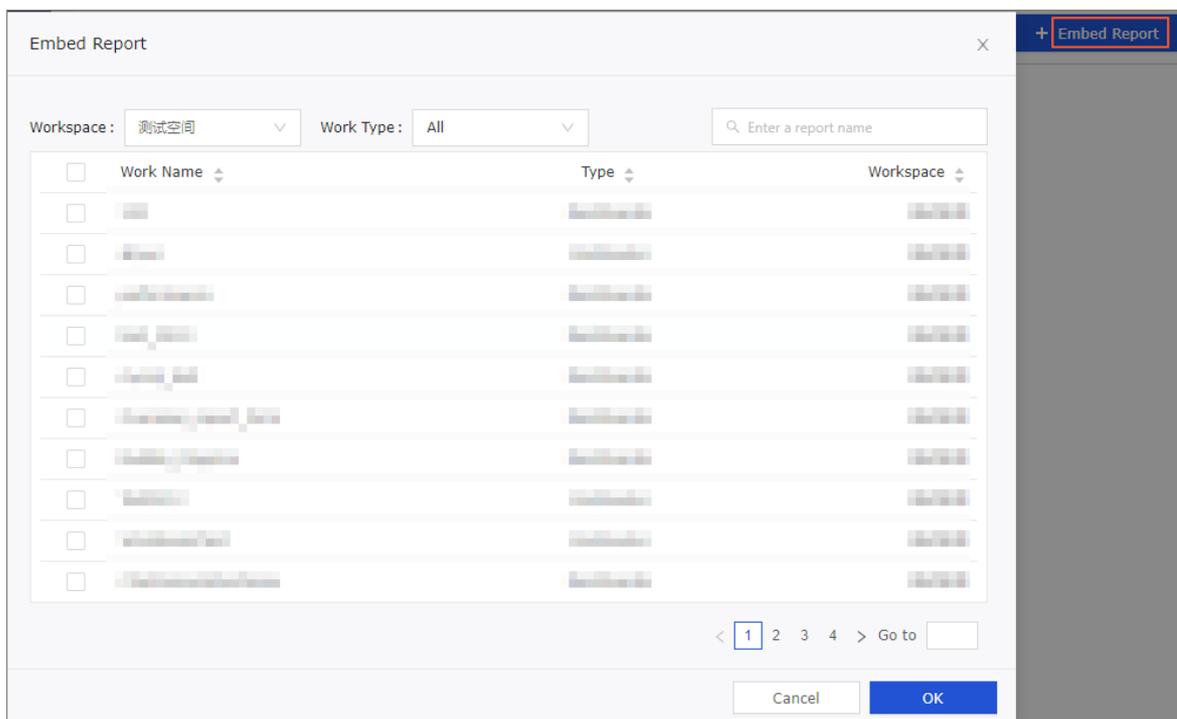
For more information about the open service, see [Report embedment](#).

7.11.2. Report embedment

This topic describes how to embed third-party reports in a workspace. Data types of reports include dashboards and workbooks.

Procedure

1. [Log on to the Quick BI console](#).
2. On the homepage of Quick BI, click **Open Service** in the top navigation bar.
3. Click **Report Embedding** under **Share Center**.
4. Click **Embed Report** in the upper-right corner.
5. In the **Embed Report** dialog box, select third-party reports that you want to embed, and click **OK**, as shown in the following figure.



6. Click the  icon next to the report. In the dialog box that appears, click **OK** to delete the embedded report.

8. Graph Analytics

8.1. What is Graph Analytics?

This topic describes the scenarios and functions of Graph Analytics.

Graph Analytics is a visual analysis platform for relationship networks. It is widely used in Alibaba Group and Ant Financial for risk control, including anti-fraud, anti-theft, and anti-money laundering solutions. Graph Analytics provides solutions for multiple industries, including public security protection, taxation, customs, banking, insurance, and the Internet.

Graph Analytics is designed to cater for multi-source data integration, computing applications, visual analytics, and intelligent businesses. Based on relationship networks, Graph Analytics can visualize the properties of objects and reveal relationships among objects.

Graph Analytics provides features including relationship networks, search networks, intelligent networks, information cubes, intelligent judgment, collaboration and sharing, and dynamic modeling. It visualizes data and integrates machine computing with human cognition. This allows you to gain insight into large amounts of data and obtain information and knowledge directly and efficiently.

8.2. Quick Start

8.2.1. Log on to Administration Console of Graph Analytics

Administration Console is the data configuration platform for Graph Analytics. In Administration Console, you can configure data sources, objects, links, events, and other advanced configuration items. Before you use Graph Analytics, you must log on to Administration Console to perform the related configurations.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

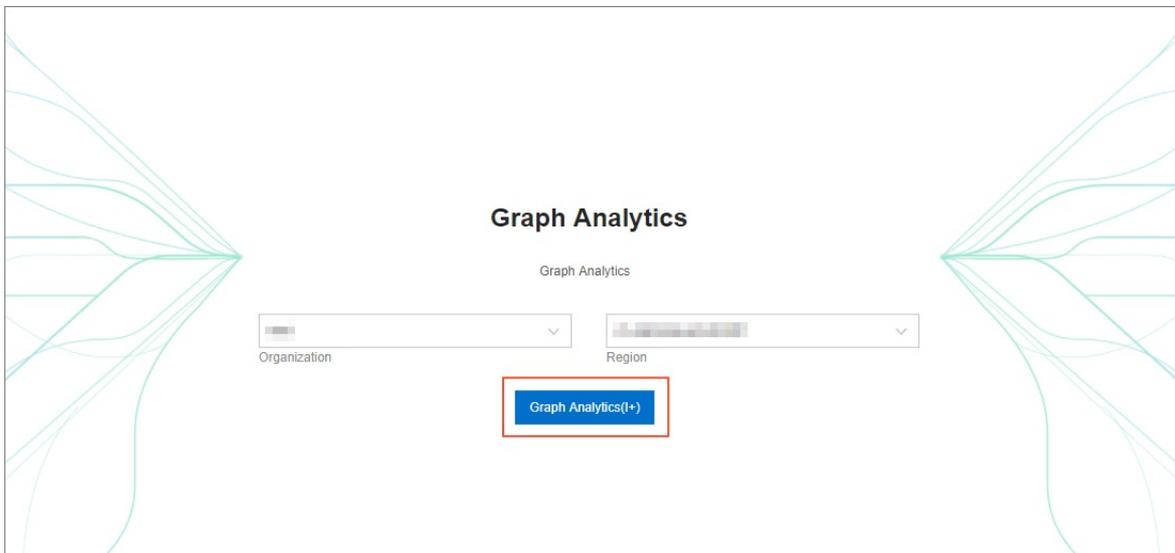
Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

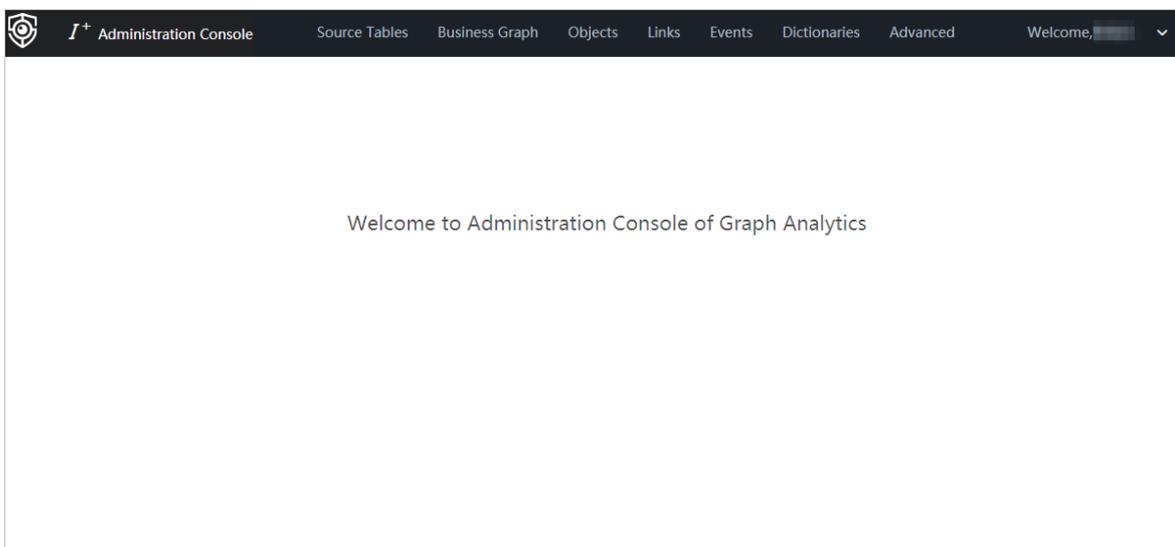
3. Click **Login**.

4. In the top navigation bar, choose **Product > Big Data > Graph Analytics**.
5. On the Graph Analytics page that appears, specify **Organization** and **Region**, and click **Graph Analytics(I+)** to redirect to the homepage of Analytics Workbench.



 **Note** You cannot use the root organization for the redirection.

6. Move your pointer over the username in the upper-right corner and select **Administration Console** to redirect to Administration Console of Graph Analytics.



8.2.2. Create data sources

Before you perform a relationship analysis, you must integrate data that you want to analyze, typically databases, into Graph Analytics. These databases will be used as data sources. In Graph Analytics, every data source is unique and can be added only once.

Prerequisites

- You have an account and password for Graph Analytics and are authorized to perform operations on Administration Console of Analytics Workbench.
- You have obtained the IP address, user name, password, port number, and other information of an

accessible data source.

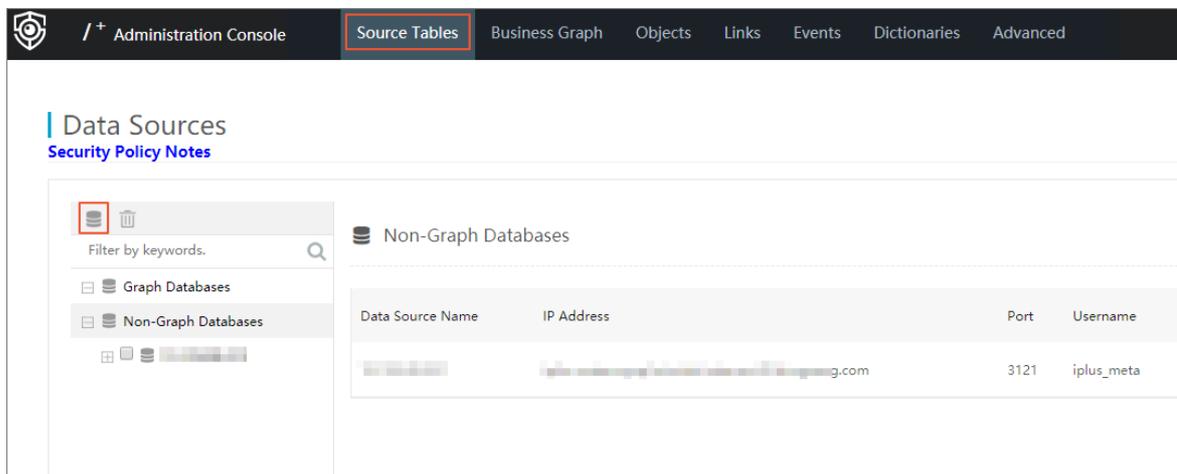
Context

A data source is one of the entries for new objects, links, and events. It is also the only entry for objects, links, and events to map to a data table. Objects, links, and events that are created on the **Object Information**, **Link Information** and **Event Information** pages are logical business objects, links, and events without mappings.

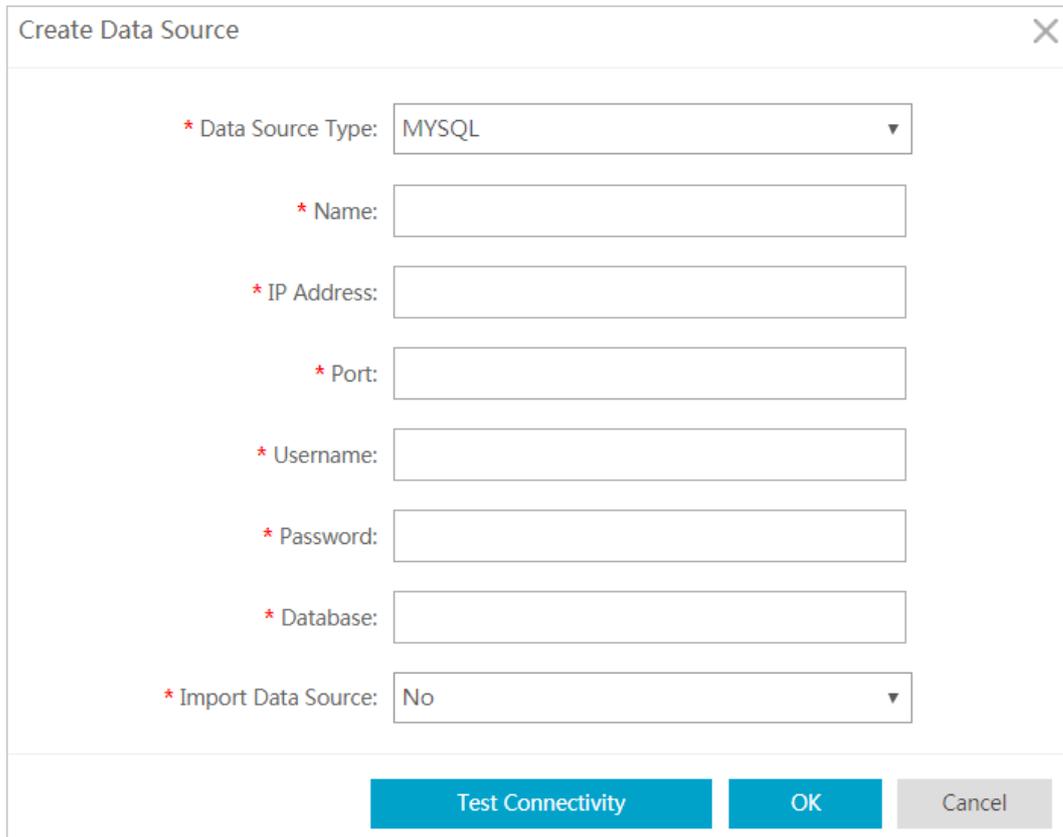
Note Objects, links, and events created for the data source take effect only after you log on again.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Source Tables** to redirect to the **Data Sources** page.



3. On the **Data Sources** page, click the **Create Data Source** icon in the left-side navigation pane. The **Create Data Source** dialog box appears.



4. In the **Create Data Source** dialog box that appears, specify the data source parameters as needed. Data source parameters

Parameter	Description
Data Source Type	The data source type you want to select. Supported data source types include: MYSQL , ORACLE , RDS , and GREENPLUM .
Name	The name of the data source. It can be user-defined.
IP Address	The IP address or domain name of the data source.
Port	The port number of the data source.
Group	The department or group to which the data source belongs.
Username and Password	The username and password that are used to connect to the data source.
Database	The name of the database that functions as the data source.
Import Data Source	If Data Source Type is not set to ORACLE , you must specify this parameter. You can import data into Analytics Workbench only after you have specified this parameter.

5. After you have configured the preceding parameters, click **Test Connectivity** to check whether the data source can be connected.

If the data source is connected properly, a message appears, indicating that the test succeeded. If the data source cannot be connected, check whether the information is correct and the data source itself is functioning properly.

6. After the connectivity test is confirmed as successful, click **OK**.

8.2.3. Create OLEP models for tables

After you add a data source, you must create object, link, event, and property (OLEP) models for the tables in the data source as needed. Before you configure OLEP tables, prepare the tables for which you will create OLEP models, the columns of each table, and the business models you want to configure. Referenced tables cannot be deleted.

Prerequisites

You have created an accessible data source.

Context

OLEP models include the following types of mappings: table-to-object mappings, table-to-link mappings, and table-to-event mappings. You can add objects, links, and events when you create OLEP models. After that, you can view and configure these objects, links, and events on the **Object Information**, **Link Information**, and **Event Information** pages, respectively. You can configure these items to reflect your business semantics. An OLEP table serves as a source for configuring objects, first-degree links, and events. A table can be mapped to multiple objects, links, and events.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click **Source Tables** in the top navigation bar. The **Data Sources** page appears.
3. Click a data source in the left-side navigation pane. The data source details are displayed on the right side of the page.
4. Click the **Not Added** tab. All tables that have no OLEP models are displayed.

You can search for a table quickly and accurately by specifying the Table Name, Table Description, or Table Group parameter.

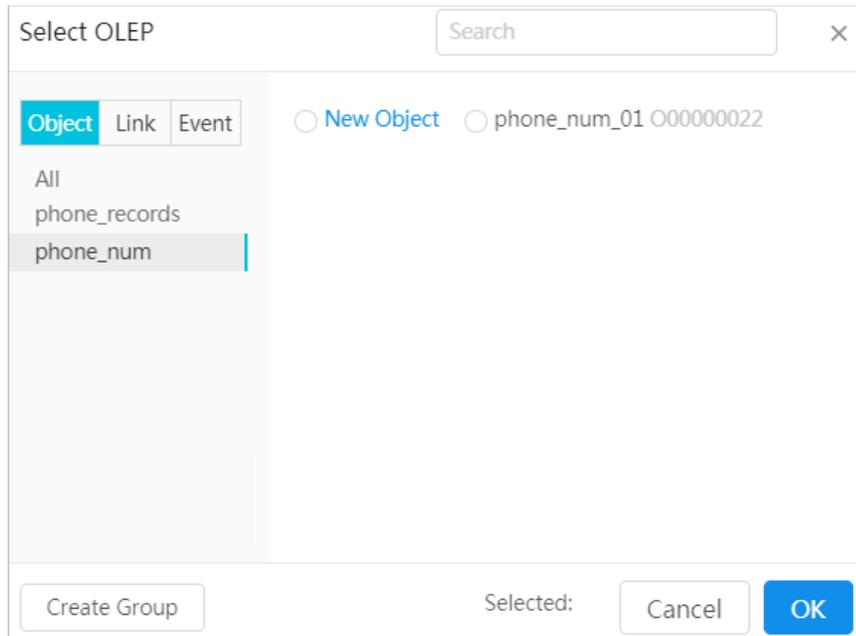
Tables that have no OLEP models

The screenshot shows the 'Data Sources' page for a data source named 'iplus_...'. It includes a 'Data Connection Information' section with fields for IP Address, Port (3306), Username, Password, Data Source Type (MYSQL), Database (iplus_...), Import Data Source (No), and Network Type (Classic Network). Below this is a tabbed interface with 'Not Added' selected. A search bar is present with fields for Table Name, Table Description, and Table Group, and a Search button. A table lists two tables: 'cust_login_info_tmp' and 'cust_regist_info_tmp', both with 'Add to OLEP' buttons in the Actions column.

Table Name	Table Description	Created Links/Objects	Table Group	Table Ro...	Last Updated At	Actions
cust_login_info_tmp				56351		Add to OLEP
cust_regist_info_tmp				2454		Add to OLEP

5. Select a table, and then click **Add to OLEP** in the Actions column. The **Select OLEP** dialog box appears.

Select OLEP dialog box



The **Select OLEP** dialog box contains the **Object**, **Link**, and **Event** tabs which are used to create mappings to objects, links, and events, respectively. For more information about how to create a mapping to an object, link, or event, see [Step 6](#), [Step 7](#), and [Step 8](#).

If no existing object groups, link groups, or event groups meet your requirements, click **Create Group** to add an object group, link group, or event group.

6. Map the table to an object.

- i. Select **New Object** or an existing object and then click **OK**. The **Map to Object** dialog box appears.

Map to a newly created object

Map to Object
✕

* Object Name: Group:

Property ID	Table Column	* Property Name	* Primary Key	Mapping <input checked="" type="checkbox"/>
O00000027P0001	<input type="text" value="identity_card"/>	<input type="text" value="identity_card"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
O00000027P0002	<input type="text" value="name"/>	<input type="text" value="name"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
O00000027P0003	<input type="text" value="phone_num"/>	<input type="text" value="phone_num"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

< 1 >

Cancel OK

Map to an existing object

Map to Object
✕

Object Name: Group: Add Property

Property ID	Table Column	Property Name	Primary Key
O00000022P0001	<input type="text" value="identity_card"/>	<input type="text" value="identity_card"/>	<input type="checkbox"/>
O00000022P0002	<input type="text" value="name"/>	<input type="text" value="name"/>	<input type="checkbox"/>
O00000022P0003	<input type="text" value="phone_num"/>	<input type="text" value="phone_num"/>	<input checked="" type="checkbox"/>

< 1 >

Cancel OK

ii. Configure the parameters as needed.

- **New object:** Configure the parameters based on [Parameters used to map the table to a new object](#).
- **Existing object:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is greater than the number of existing properties of the object, you can click **Add Property** to add new properties.

Parameters used to map the table to a new object

Field	Description
Object Name	The user-defined object name. It must be unique.
Group	The object group to which the object belongs. All available object groups are displayed in the drop-down list.
Property Name	The name of an object property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, property names are displayed instead of the actual table columns that are mapped to the properties.
Mapping	Indicates whether to enable the property mapping.
Primary Key	The property that is used as a primary key. Each primary key uniquely identifies an object. You must configure one or more properties as primary keys for each object. You must enable Mapping for primary keys.

iii. Click **OK**.

7. Map the table to a link.

- i. Click the **Link** tab. All first-degree links to which the current table has been mapped are displayed.
- ii. Select **New Link** or an existing link and then click **OK**. The **Map to Link** dialog box appears.

Map to a newly created link

Map to Link ✕

* Link Name: Group:

* Source Object: * Target Object:

Basic Information

Property ID	Table Column	* Property Name	Mapping <input checked="" type="checkbox"/>
L00000016P0001	<input type="text" value="callee_num"/>	<input type="text" value="callee_num"/>	<input checked="" type="checkbox"/>
L00000016P0002	<input type="text" value="caller_num"/>	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>

< 1 >

^ Source Property Mapping

SourceObject Property:phone_num_01 - phone_num * Link Property:

^ Target Property Mapping

TargetObject Property:phone_num_01 - phone_num * Link Property:

Map to an existing link

Map to Link ✕

Link Name: Group:

Source Object: Target Object: Create Property

Basic Information

Property ID	Table Column	Property Name
L00000014P0001	<input type="text" value="caller_num"/> *	<input type="text" value="caller_num"/>
L00000014P0002	<input type="text" value="callee_num"/> *	<input type="text" value="callee_num"/>

< 1 >

^ Source Property Mapping

SourceObject Property:phone_num_01 - phone_num * Link Property:

^ Target Property Mapping

TargetObject Property:phone_num_01 - phone_num * Link Property:

iii. Configure the parameters as needed.

- **New link:** Configure the parameters based on [Parameters used to map the table to a new link](#).
- **Existing link:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is more than the number of existing link properties, you can click **Add Property** to add new link properties.

Parameters used to map the table to a new link

Field	Description
Link Name	The user-defined link name. It must be unique.
Group	The link group to which the link belongs. All available link groups are displayed in the drop-down list.
Source	The source object of the link. You can select an object from the drop-down list. The Source Property Mapping parameter is displayed only after you configure the Source Object parameter.
Target	The target object of the link. You can select an object from the drop-down list. The Target Property Mapping parameter is displayed only after you configured the Target Object parameter.
Property Name	The name of a link property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, the Property Name values are displayed as the table header in Details on the Graph page.
Switch	Indicates whether to enable the property mapping.
Link Property in Source Property Mapping	The link property to which a primary key property of the source object is mapped.
Link Property in Target Property Mapping	The link property to which a primary key property of the target object is mapped.

iv. Click **OK**.

8. Map the table to an event.

- i. Click the **Event** tab. All events to which the current table has been mapped are displayed.
- ii. Select **New Event** or an existing event and then click **OK**. The **Map to Event** dialog box appears.
 Map to a newly created event

Map to Event ✕

* Event Name : Group:

Basic Information

Property ID	Table Column	* Property Name	* Primary Key	Mapping <input type="checkbox"/>
E00000014P0001	<input type="text" value="callee_num"/>	<input type="text" value="callee_num"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
E00000014P0002	<input type="text" value="caller_num"/>	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

< 1 >

Primary Key Mappings of Correlated Objects Add Mapping

<input type="text" value="phone_num O00000022"/>	phone_num(O00000022P0003) :	<input type="text" value="caller_num E00000014P00..."/>
<input type="text" value="phone_num O00000022"/>	phone_num(O00000022P0003) :	<input type="text" value="callee_num E00000014P00..."/>

Map to an existing event

Map to Event ✕

Event Name : Group: Add Property

Basic Information

Property ID	Physical Table Field	Property Name	Primary Key
E00000014P0001	<input type="text" value="caller_num"/> *	<input type="text" value="callee_num"/>	<input type="checkbox"/>
E00000014P0002	<input type="text" value="callee_num"/> *	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>

< 1 >

^ Primary Key Mappings of Correlated Objects

phone_num(O00000022P0003) :

phone_num(O00000022P0003) :

OK

iii. Configure parameters as needed.

- **New event:** Configure the parameters based on [Parameters used to map the table to a new event](#).
- **Existing event:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is greater than the number of existing event properties, you can click **Add Property** to add new event properties.

Parameters used to map the table to a new event

Field	Description
Event Definition Name	The user-defined event name. It must be unique.
Group	The event group to which the event belongs. All available event groups are displayed in the drop-down list.
Property Name	The name of an event property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, the Property Name values are displayed as the table header in Details on the Graph page.
Switch	Indicates whether to enable the property mapping.
Primary Key	The property that is used as a primary key. Each primary key uniquely identifies an event. You must configure one or more properties as primary keys for each event. Switch must be turned on for the properties that are configured as primary keys.
Map Primary Keys to Correlated Objects	The mappings between the primary keys of correlated objects and the event properties. Two or more correlated objects are required. You can click Add Mapping to add more necessary mappings between the primary keys of correlated objects and the event properties. You must enable Mapping for the event properties to which the primary keys of the correlated objects are mapped.

iv. Click **OK**.

9. After you create OLEP models for the data tables, click the **Added to OLEP** tab to check the results.

8.2.4. Add OLEP table columns

If a data table has been mapped to an object, link, or event, and the table still has unoccupied columns (columns that are not correlated with any object, link, or event), you can add these columns to the existing mappings as needed.

Prerequisites

A data table has been mapped to an object, link, or event, but the table still has unoccupied columns.

Context

Before you configure the OLEP table columns, sort out the columns for which you will create OLEP models and data types of the columns, especially the time columns.

Procedure

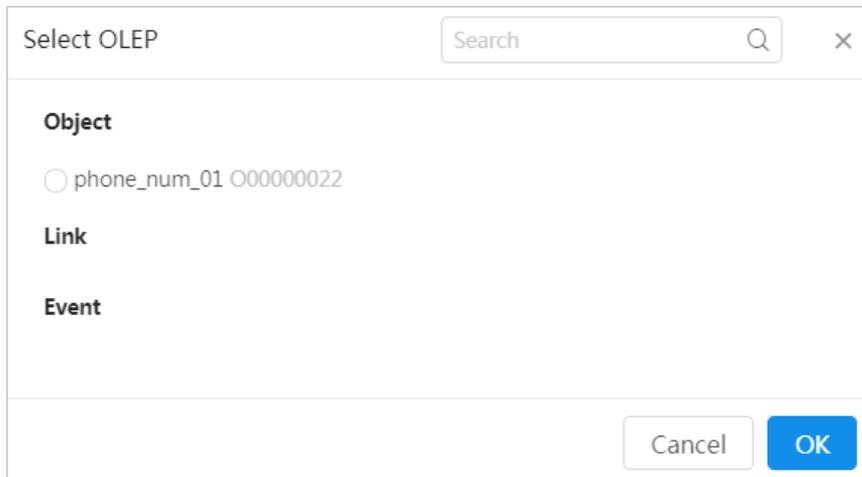
1. Log on to Administration Console of Graph Analytics.
2. Click **Source Tables** in the top navigation bar.
3. On the **Data Sources** page, click a data source in the left-side navigation pane, and then click the table to which you want to add the columns.
4. Click the **Columns Not Added** tab in the right-side area. In the **Columns Not Added** tab that appears, click **Add** in the Actions column.

Columns displayed in the **Columns Not Added** tab are not mapped to any object, link, or event.

5. In the **Select OLEP** dialog box that appears, select the object, link, or event to which the columns map, and then click **OK**.

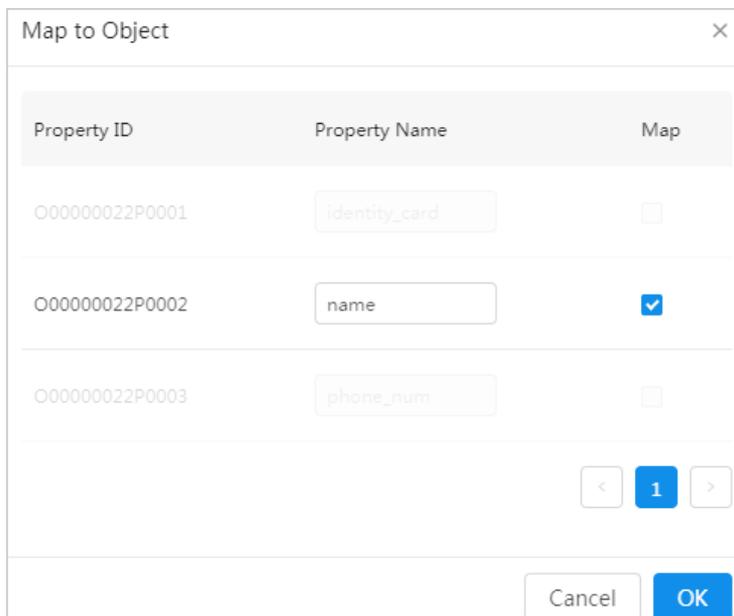
The **Select OLEP** tab only displays the objects, links, and events that have mapped to the current data table.

The following example demonstrates how to add columns to an object.



6. In the **Map to Object** dialog box that appears, select the object properties that map to the columns you want to add.

Object properties that have been mapped to the current data table are dimmed and cannot be operated.



7. After you configure the preceding parameters, click **OK**.

8.2.5. Configure object properties and business parameters

After you add an object, you need to configure the business parameters of the object based on your requirements so that you can view and analyze the object in Analytics Workbench.

Prerequisites

- You have created a data source. For more information, see [Create data sources](#).
- You have configured mappings between tables and objects, links, or events. For more information, see [Create OLEP models for tables](#).

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click **Objects** on the top of the page.
3. In the left-side navigation pane of the **Object Information** page, click the name of the object you want to configure and then click the **Property Information** tab on the right side.

If this object has been mapped to a physical table in the data source, the **Property Information** tab displays the **Data Source** and **Table** information.

The screenshot shows the 'Property Information' tab for an object named 'SV_Account'. The 'Data Source' is 'skyview' and the 'Table' is 'demo_tranx_id'. The 'Basic Information' section contains a table with the following data:

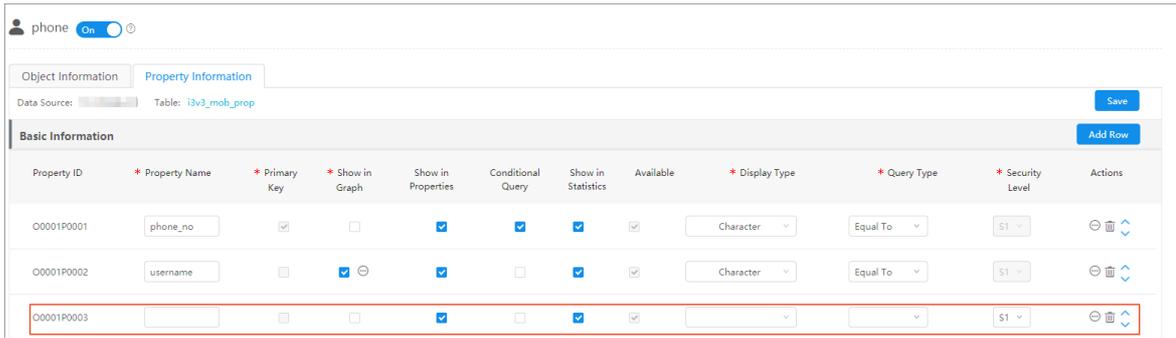
Property ID	* Property Name	* Primary Key	* Show in Graph	Show in Properties	Conditional Query	Show in Statistics	Available	* Display Type	* Query Type	* Security Level	Actions
O00000072P0001	TRANX_ID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Character	String Equal	S1	<input type="button" value="edit"/> <input type="button" value="delete"/> <input type="button" value="refresh"/>
O00000072P0002	BRANCH_NAV	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Character	String Equal	S1	<input type="button" value="edit"/> <input type="button" value="delete"/> <input type="button" value="refresh"/>
O00000072P0003	NAME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Character	String Equal	S1	<input type="button" value="edit"/> <input type="button" value="delete"/> <input type="button" value="refresh"/>

Below the table, there are sections for 'Advanced' (Add Image To, Logical Relation of Authorized Properties), 'Location Settings' (Add Location), and 'Trajectory Settings' (Add Trajectory).

4. Specify parameters in the **Basic Information** section.

Click **Add Row** to add a property in **Basic information**.

[Parameters in Basic Information](#) describes the parameters in **Basic Information**.



Parameters in Basic Information

Parameter	Description
Property ID	The ID of a property. It is automatically generated.
Property Name	The property name that is displayed on Analytics Workbench. Enter a name that describes your business.
Primary key	<p>You must select one or more primary keys for the properties when you add an object for the first time. After the configuration is complete, you cannot modify or delete the primary keys.</p> <p>If you add a node in Analytics Workbench, you must enter the physical table column mapped by the primary key properties. For example, if the ID card number is the primary key, you need to enter the ID card number when you add an ID card node in Analytics Workbench.</p>
Show in Graph	<p>If you select this parameter, Property Name of this property will be displayed in Graph in Analytics Workbench together with the object node. Otherwise, Property Name will not be displayed. For example, if you select the ID card number, this number will be displayed in Graph together with the ID card object. If you select the name property at the same time, the name and the ID card number will be displayed together with the ID card object.</p> <p>If you select this parameter for a specific property, a bubble icon appears next to the option. You can click the bubble icon to specify whether to show the Property Name in Graph. For example, if you select this parameter for the ID card number, and choose to display Property Name in Graph, the ID card property displayed in Graph will be ID card number: xxxxxx.</p>
Show in Properties	If you select this parameter for a property, the property will be displayed on the Details tab and the Property tab in the right-side pane of the Graph page in Analytics Workbench. Otherwise, the property will not be displayed.
Conditional Query:	If you select this parameter for a property, you can query the object based on this property in Target Object when you perform an analysis on the Graph page.
Show in Statistics	If you select this parameter for a property, the property is displayed on the Statistics tab in the right-side pane of the Graph page in Analytics Workbench. If you do not select this parameter, the property is not displayed in Analytics Workbench.

Parameter	Description
Available	<p>If you select this parameter for a property, the property takes effect and is displayed in Analytics Workbench. This parameter must be selected for primary key properties.</p> <p>If any of the following parameters has been selected for the property: Primary Key, Show in Graph, Show in Properties, Conditional Query and Show in Statistics, the Available parameter will be automatically selected for a property. The Available parameter will be automatically cleared if you clear all the preceding parameters.</p>
Display Type	<p>After you configure this parameter, the property is displayed on the Details tab and the Property tab in the right-side pane of the Graph page in Analytics Workbench based on the selected type.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #add8e6;"> <p> Note To display a property in the format of Dictionary, you need to configure a dictionary first.</p> </div>
Query Type	<p>The data type that is supported in the query condition of a property. If you select Dictionary for Display Type, you must select Dictionary Option for Query Type.</p>
Security Level	<p>The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.</p>
Search Item Configuration	<p>Click the  icon in the Actions column corresponding to a property. Specify the following parameters:</p> <ul style="list-style-type: none"> ○ Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. ○ Default Query Condition Settings: the default condition used for an object query. If other properties are used as conditions for a query, this condition is also used by default. ○ Authorization Code: After the authorization code function is enabled, only the authorized users can access this property. ○ Derived Property: After a property is set as a derived property, it is automatically generated based on other properties. Configure the method in which the field is generated as needed.
Default Query Condition Settings	
Authorization Code	
Derived Field	
Delete	<p>If a property is no longer used, you can click the  icon to delete this property.</p>
Sort order	<p>The Move Up and Move Down arrows are used to adjust the order of properties that are displayed in Analytics Workbench.</p>

5. (Optional) If you need to add multiple properties, you can refer to the preceding steps to add more properties.

6. (Optional) Specify parameters in **Advanced**.

[Parameters in Advanced](#) describes the parameters in **Advanced**.

Parameters in Advanced

Parameter	Description
Add Image To	The avatar of the object that is displayed in Graph . Select a property of the object, and then specify the URL of the image and the suffix of the image. Add Image To allows you to specify a combination of the prefix, the property, and the suffix. The prefix is the URL of the image, and the suffix is the image format.
Logical Relation of Authorized Properties	The logical relationship between the authorization codes of properties in each record: <ul style="list-style-type: none"> ◦ AND: The current record is visible to the users who meet all authorization code conditions of the properties in this record. ◦ OR: The current record is visible to the users who meet any one authorization code condition of the properties in this record.

7. Click **Save**.

8.2.6. Configure link properties and business parameters

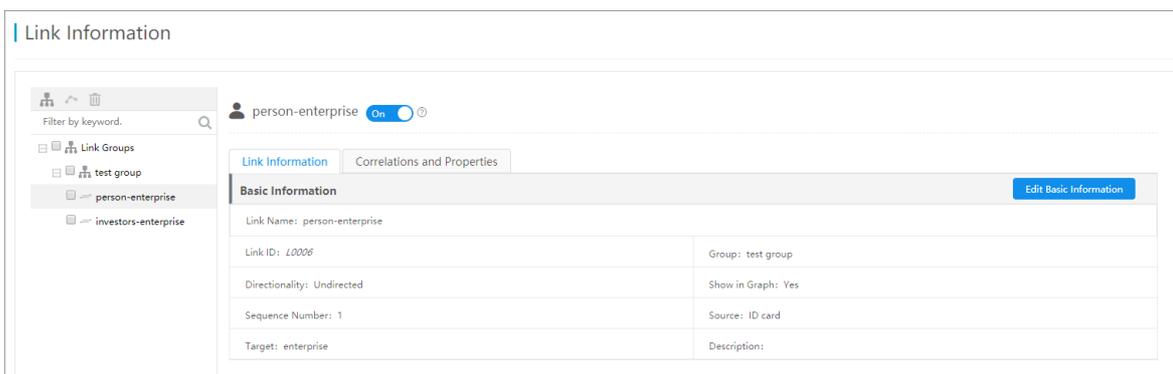
After a first-degree link is added, you need to configure the properties and business parameters of the link based on your business requirements, so that you can view and apply this link in Analytics Workbench. This topic describes how to configure the properties and business parameters of a first-degree link.

Prerequisites

A first-degree link is created. For more information, see [Create a first-degree link](#).

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.
3. In the left-side navigation pane, click the link group that contains the first-degree link you want to configure, and then click the target link.



4. On the right side of the page, click the **Correlations and Properties** tab.

On the **Correlations and Properties** tab, if the link is mapped to a data table in the data source, **Data Source** and **Table** are displayed in the property information above the Basic Information section. You can click the table name to redirect to the table page.

5. Configure the parameters in the **Basic Information** section.

If you need to add a property for a link in the **Basic Information** section, click **Add Row**.

Note

Link-Correlated Property in the **Source Property Mapping** section and Link-Correlated Property in the **Target Property Mapping** section must be different and both of them must be configured. Therefore, a link must have two or more properties.

[Parameters in the Basic Information section](#) describes the parameters in the **Basic Information** section.

Parameters in the **Basic Information** section

Parameter	Description
Property ID	The ID of a property. It is automatically generated.
Property Name	The name of a property. If you select Available for a property, the name of the property is displayed in Analytics Workbench.
Unique ID	Allows you to define the logical primary key of a data table.
Show in Details	If you select this parameter for a property, the property is displayed on the Details tab in Analytics Workbench.
Conditional Query	If you select this parameter for a property, you can query a link based on the link type specified in Link Type when you perform analysis on the Graph page in Analytics Workbench.
Statistics	If you select this parameter for a property, the property is displayed on the Statistics tab in the right-side pane of the Graph page in Analytics Workbench. If you do not select this parameter, the property is not displayed in Analytics Workbench.

Parameter	Description
Available	<p>If you select this parameter for a property, the property takes effect and is displayed in Analytics Workbench.</p> <p>If one of the following parameters is selected for the property, the Available option is automatically selected for a property: Unique ID, Show in Details, Conditional Query, and Show in Statistics. If all of the preceding parameters are cleared for a property, the Available parameter is automatically cleared for the property.</p>
Display Type	<p>After you set the display type, the property is displayed on the Property tab in the right-side pane of the Graph page in Analytics Workbench based on the selected type.</p> <div style="background-color: #e0f2f1; padding: 5px; border: 1px solid #ccc;"> <p> Note If you want to display a property in the Dictionary format, configure a dictionary first.</p> </div>
Query Type	<p>The data type that is supported in the query condition of a property. If you select Dictionary for Display Type, you must select Dictionary Option for Query Type.</p>
Security Level	<p>The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.</p>
Search Item Configuration	<p>Find the property and click the More icon  in the Actions column. Configure the following parameters:</p> <ul style="list-style-type: none"> ◦ Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. ◦ Default Query Condition Settings: the default condition used for a link query. If other properties are used as conditions for a query, this condition is also used by default. ◦ Authorization Code: After the authorization code function is enabled, only authorized users can access this property. ◦ Derived Property: After a property is configured as a derived property, it can be generated automatically based on other properties. Configure the method in which the column is generated based on your needs.
Default Query Condition Settings	
Authorization Code	
Derived Field	
Delete	<p>If a property is no longer used, you can click the Delete icon  to delete this property.</p>
Sort order	<p>The Move Up and Move Down arrows are used to adjust the order of properties that are displayed in Analytics Workbench.</p>

6. Set **Link-Correlated Property** in the **Source Property Mapping** and **Target Property Mapping** sections.

These configurations are related to Source Object and Target Object of the link.

Use **Source Property Mapping** as an example. **Source Object Correlated Property** and **Link-Correlated Property** must be mapped to the same column in the same table. The **Source Object Correlated Property** parameter is the primary key property of the source object, which is automatically loaded based on the **Source Object** parameter. For the **Link-Correlated Property** parameter, you must select the link property that is mapped to the same column in the same table as the **Source Object** primary key.

Set **Target Property Mapping** in the same way you set **Source Property Mapping**.

- (Optional) Configure parameters in the **Advanced**, **Accumulative Statistics Settings**, and **Link Weight Settings** sections based on your requirements.

For more information about the key parameters, see [Key parameters](#).

The screenshot shows a configuration interface with three main sections:

- Advanced:** Contains fields for Chronological Time Property, Time Property for Behavior Analysis, Linked Times (dropdown), Details Sorting Property (dropdown), Logical Relation of Authorized Properties (dropdown), and Link Name Definition (Link Name dropdown, person-enterprise text).
- Accumulative Statistics Settings:** Includes a blue 'Set Linked Times' button and a grey 'Add Statistical Condition' button.
- Link Weight Settings:** Includes a checkbox for 'Configure Link Weight', a 'Weight Property' dropdown, 'Sequence' radio buttons (Ascending selected, Descending), 'Weight Thresholds' input fields, and a label 'Weight Segments from Low to High (-∞, +∞)'.

Key parameters

Section	Parameter	Description
Advanced	Chronological Time Property	The link properties involved in chronological analysis. From the drop-down list, select one or more link properties with the query type of time.
	Time Property for Behavior Analysis	The link properties involved in behavior analysis. From the drop-down list, select one or more link properties with the query type of time.
	Linked Times	The number of the same values that are counted for a specific property in a link. The total number is displayed as the number of link occurrences. The Linked Times parameter is used as the default setting to filter link types. Assume that two lines of calls from user A to user C are displayed in the call log. The number of calls from user A to user C is displayed as two in the analysis results.
	Details Sorting Property	The default property by which the returned behavior details are sorted.

Section	Parameter	Description
Accumulative Statistics Settings	None	Used to perform logical statistics for link properties of which the query type is numeric range. The logical statistics operations include top, equal sign (=), and greater-than-or-equal-to sign (≥). This configuration is suitable for business scenarios where statistics filter is required for link query results. The Linked Times parameter is used to filter records in the link query results. You can add statistical conditions to filter the link properties with the query type of numeric range.
Link Weight Settings	None	You can specify a link property with the query type of numeric range and calculate the link weight based on the numeric range specified for the link property.

- After you modify the parameter configurations, click Save. A message appears, indicating that the modifications are saved.

8.2.7. Configure event properties and business parameters

After you have created an event, you must configure the event properties. Event properties are critical to an event. You can configure event properties, and associate the properties with objects on the Property Information tab.

Prerequisites

You have obtained an account and a password for Graph Analytics and you are granted the required **event permissions**.

Procedure

- [Log on to Administration Console of Graph Analytics](#).
- Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
- Click an event in the left-side navigation pane, and click the **Property Information** tab on the right side of the page. The **Property Information** tab appears.
- Set the required event parameters.

Click **Add Row** to add a property for a link in **Basic information**.

The required parameters are displayed in the **Basic Information** and **Set Mappings Between Correlated Objects and Properties** areas. [Required event property parameters](#) describes the event parameters.

 **Note** To save the property information, you must set all the required parameters.

Required event property parameters

Category	Parameter	Description
	Property ID	The ID of a property. It is automatically generated.

Category	Parameter	Description
Basic information	Property Name	The property name that is displayed on Analytics Workbench. We recommend that you enter a name that describes the business type.
	Primary key	Each primary key uniquely identifies an event. A property cannot be deleted after it has been configured as a primary key.
	Show in Graph	<p> Note For each event, you must select one or more properties you want to display in the graph.</p> <p>If you select this parameter, the Property Name of this property will be displayed in Graph in Analytics Workbench together with the object node. Otherwise, the property is not displayed. For example, if this parameter is selected for the ID card property of an event, the ID card number will be displayed in Graph with the event. If the name property is also selected, the name and the ID card number will be displayed with the event.</p> <p>If you select this parameter for a specific property, a bubble icon appears next to the option. You can click the bubble icon to specify whether to show the Property Name in Graph. For example, if you select this parameter for an ID card number, and choose to display the Property Name in Graph, the ID card property displayed in Graph will be ID card number: xxxxxx.</p>
	Show in Properties	If you select this parameter for a property, the property will be displayed in the right-side pane on the Graph page of Analytics Workbench. To view the property, choose Properties > Event Properties .
	Element Identifier	The element identifier of a property. Set this parameter based on the actual property.
	Conditional Query	If you select this parameter for a property, the event can be queried based on this property in Link Type on the Graph page of Analytics Workbench when you perform a relationship analysis.
	Statistics	If you select this parameter for a property, the event will be displayed in the right-side pane on the Graph page of Analytics Workbench. To view the property, choose Statistics > Event Distribution .
	Available	<p>If you select this parameter for a property, the property takes effect and can be displayed on the Graph page of Analytics Workbench.</p> <p>This parameter is selected by default and cannot be changed.</p>

Category	Parameter	Description
	Display Type	The format in which a property is displayed in the right-side pane of the Graph page on Analytics Workbench. Set this parameter as needed.
	Query Type	The data type that is supported in the query condition of a property.
	Security Level	The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.
	Search Item Configuration	Click the More icon  in the Actions column that corresponds to a property. Configure the following parameters: <ul style="list-style-type: none"> Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. Default Query Condition Settings: The default condition used for a link query. If other properties are used as conditions for a query, this condition is also used by default. Authorization Code: After the authorization code function is enabled, only authorized users can access this property. Derived Property: After a property is set as a derived property, it can be generated automatically based on other properties. Configure the method in which the column is generated based on your needs.
	Default Query Condition Settings	
	Authorization Code	
	Derived Field	
	Delete	If a property is no longer used, you can click the Delete icon  to delete this property.
Sort order	The Move Up and Move Down arrows are used to adjust the order of properties displayed in Analytics Workbench.	
Set Mappings Between Correlated Objects and Properties	Add Correlated Object	Adds a mapping between an object and the event. One event must have two or more objects mapped to it. Click Add Correlated Object to add a correlated object, and configure the mapping between the event and the primary keys of the object you have added.

5. (Optional)After you have specified the required parameters, you can specify the optional parameters as needed.

The optional parameters are included in the **Advanced** and **Display Settings** areas. The optional parameters are described in [Optional event property parameters](#).

 **Note** Location Settings is currently not supported.

Optional event property parameters

Section	Parameter	Description
Advanced	Behavior Property	The properties based on which a behavior analysis is performed.
	Default Details Sorting Property	The property by which the details are sorted.
	Logical Relation of Authorized Properties	<p>The logical relationship between the authorization codes of properties in each record.</p> <ul style="list-style-type: none"> ◦ AND: The current record is visible only to the users who meet all authorization code conditions of the properties in this record. ◦ OR: The current record is visible to the users who meet any one authorization code condition of the properties in this record.
Display Settings	Enable Display	Indicates whether to show the event details.
	Group-by Properties	The property based on which events are aggregated. For example, aggregate T ra v el E vents into a folder based on the T rain N umber property.

- After the configurations are complete, click **Save** in the upper-right corner. A success message is displayed after the modifications are saved.

An event is automatically enabled after its properties are saved.

8.2.8. Log on to Analytics Workbench

Analytics Workbench is a data analysis platform of Graph Analytics. After you configure relevant data in Administration Console of Graph Analytics, you can perform data analysis in Analytics Workbench.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

- In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
- Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

Note When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, choose **Product > Big Data > Graph Analytics**.
5. On the Graph Analytics page that appears, specify **Organization** and **Region**, and click **Graph Analytics(+)** to go to the homepage of Analytics Workbench.

Note You cannot use the root organization for the redirection.

8.2.9. Create an analysis

After you log on to Analytics Workbench, you must create an analysis and add objects as nodes before you perform an analysis.

Prerequisites

- You have obtained an account and a password with the permission to perform graphic operations.
- You have created source tables, objects, links, and events.
- You have obtained data in the tables that have been mapped to the primary keys of the objects you want to analyze. You can query the tables in the database to obtain the data.

Procedure

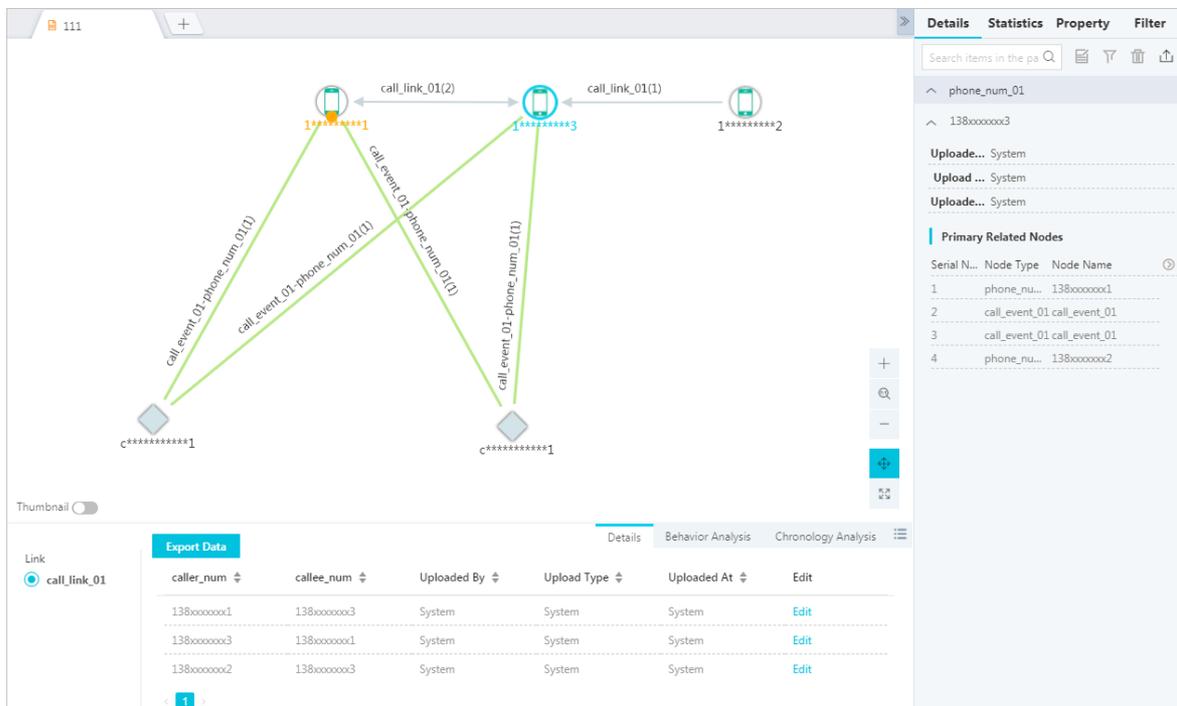
1. [Log on to Analytics Workbench](#).
2. Click **Create Analysis**. A **Temporary Analysis** tab appears.
3. Click **Add** in the toolbar and then click the blank space, or right-click the blank space and select **Add Node**. Specify parameters in the **Add Node** dialog box that appears.

[Parameters to add a node](#) describes the parameters.

Parameters to add a node

Parameter	Description
Object type	Displays all created objects. You can select an object as needed. Note Graph Analytics allows you to add compound nodes. For example, you can specify two sub-types for object type "person": ID card and passport. The person can be uniquely identified by the ID card or the passport.
Text area	Enter one or more primary key values. Separate multiple primary key values with commas (,).

- Click **OK**.
- Right-click a node that has been added, and select **Quick Extension**. The system automatically performs a link analysis based on the configured data sources, objects, links, and events, and displays the analysis results in a graph.



- Select one or more objects, links, or events. Click **Behavior Chronology** in the lower-right corner to view the information on the **Details**, **Behavior Analysis**, and **Chronology Analysis** tabs.
- Select one or more objects, links, or events. Click the  icon in the upper-right corner of the right-side pane to view the information on the **Details**, **Statistics**, **Property**, and **Filter** tabs.
- After the analysis is complete, click **Save** in the upper-right corner. In the **Save Analysis** dialog box, specify **File Name**, select a folder, and then click **OK**. A success message appears after the file has been saved.

After you have saved the analysis file, if a collaborative analysis is required, you can share this personal analysis with other members.
- Click the **Share** icon in the upper-right corner to specify the members you want to share this analysis with.

8.2.10. View analyses

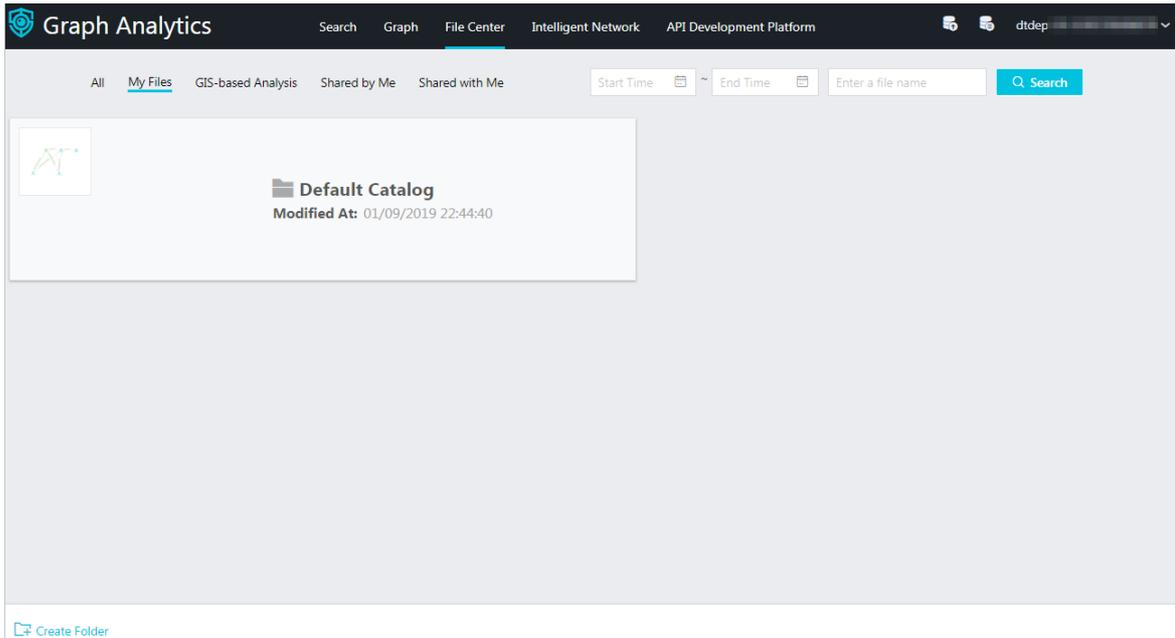
After you close the analysis file or log on to Analytics Workbench again, you can view the existing analyses or shared analyses in **File Center**.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Procedure

- [Log on to Analytics Workbench.](#)
- Click **File Center** in the top navigation bar. The **File Center** page appears.



3. Click **My Files**, **Shared by Me**, or **Shared with Me** to view the corresponding analysis files.
4. Double-click an analysis to directly open the analysis file on the **Graph** page.

8.3. Source tables

8.3.1. Data sources

8.3.1.1. Create data sources

Before you perform a relationship analysis, you must integrate data that you want to analyze, typically databases, into Graph Analytics. These databases will be used as data sources. In Graph Analytics, every data source is unique and can be added only once.

Prerequisites

- You have an account and password for Graph Analytics and are authorized to perform operations on Administration Console of Analytics Workbench.
- You have obtained the IP address, user name, password, port number, and other information of an accessible data source.

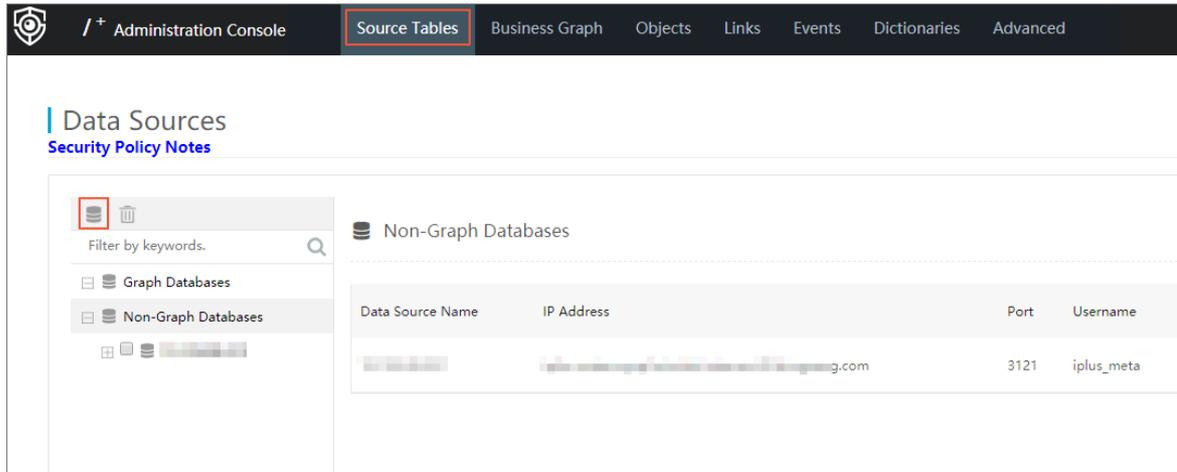
Context

A data source is one of the entries for new objects, links, and events. It is also the only entry for objects, links, and events to map to a data table. Objects, links, and events that are created on the **Object Information**, **Link Information** and **Event Information** pages are logical business objects, links, and events without mappings.

Note Objects, links, and events created for the data source take effect only after you log on again.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Source Tables** to redirect to the **Data Sources** page.



3. On the **Data Sources** page, click the **Create Data Source** icon in the left-side navigation pane. The **Create Data Source** dialog box appears.

Create Data Source ✕

* Data Source Type:

* Name:

* IP Address:

* Port:

* Username:

* Password:

* Database:

* Import Data Source:

4. In the **Create Data Source** dialog box that appears, specify the data source parameters as needed. Data source parameters

Parameter	Description
Data Source Type	The data source type you want to select. Supported data source types include: MYSQL , ORACLE , RDS , and GREENPLUM .
Name	The name of the data source. It can be user-defined.
IP Address	The IP address or domain name of the data source.

Parameter	Description
Port	The port number of the data source.
Group	The department or group to which the data source belongs.
Username and Password	The username and password that are used to connect to the data source.
Database	The name of the database that functions as the data source.
Import Data Source	If Data Source Type is not set to ORACLE , you must specify this parameter. You can import data into Analytics Workbench only after you have specified this parameter.

- After you have configured the preceding parameters, click **Test Connectivity** to check whether the data source can be connected.

If the data source is connected properly, a message appears, indicating that the test succeeded. If the data source cannot be connected, check whether the information is correct and the data source itself is functioning properly.

- After the connectivity test is confirmed as successful, click **OK**.

8.3.1.2. View data sources

Graph Analytics allows you to view all data sources in the current environment. This function helps you understand the existing data source information at any time.

Procedure

- [Log on to Administration Console of Graph Analytics.](#)
- Click **Source Tables** in the top navigation bar. The **Data Sources** page appears.
- Select a data source from the left-side navigation pane. The data source details are displayed on the right side of the page, as shown in [Data source details](#).

Data source details

The screenshot displays the 'Data source details' page. At the top, there is a header with the user 'iplus_...' and a 'Data Connection Information' section with an 'Edit Information' button. The connection details are as follows:

- IP Address: [Redacted]
- Port: 3306
- Username: [Redacted]
- Password: [Redacted]
- Data Source Type: *MYSQL*
- Database: *iplus_...*
- Import Data Source: *No*
- Network Type: *Classic Network*

Below the connection info, there are search filters for 'Table Name', 'Table Description', and 'Table Group', along with an 'Add to OLEP' button (currently 'Not Added') and a 'Search' button. A table lists the data sources:

Table Name	Table Description	Created Links/Objects	Table Group	Table Ro...	Last Updated At	Actions
cust_login_info_tmp				56351		Add to OLEP
cust_regist_info_tmp				2454		Add to OLEP

At the bottom right, there are navigation arrows and a page number '1'.

8.3.1.3. Modify a data source

Graph Analytics allows you to modify all data sources in the current environment. This function helps you adjust basic information about data sources at any time if required.

Prerequisites

To modify a data source, you must have the user password that is used to connect to the data source.

Context

You can modify all parameters of a data source except Data Source Type.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click **Source Tables** in the top navigation bar. The **Data Sources** page appears.
3. Click a data source in the left-side navigation pane, and click **Edit Information** in the Data Connection Information section on the right side of the page. Alternatively, click the parent directory of the target data source in the left-side navigation pane, and click **Modify** next to the target data source in the data source list on the right side of the page.
4. In the dialog box that appears, modify the data source parameters as needed, and enter the user **password**.
5. Click **Test Connectivity** to verify the modifications.
6. If the test is successful, click **OK**.

8.3.1.4. Delete a data source

You can delete a data source if it is no longer used.

Prerequisites

All tables in the data source are not correlated with any objects, links, or events. If a data table is correlated with an object, link, or event, remove the association first. For more information, see [Remove OLEP tables](#).

Context

Deleted data sources cannot be recovered. Exercise caution when you delete a data source.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click **Source Tables** in the top navigation bar. The **Data Sources** page appears.
3. Select one or more data sources from the left-side navigation pane, and then click the  icon in the top navigation bar.
4. In the Delete Data Sources dialog box that appears, click **OK** to delete the specified data sources.

8.3.2. OLEP tables

8.3.2.1. Create OLEP models for tables

After you add a data source, you must create object, link, event, and property (OLEP) models for the tables in the data source as needed. Before you configure OLEP tables, prepare the tables for which you will create OLEP models, the columns of each table, and the business models you want to configure. Referenced tables cannot be deleted.

Prerequisites

You have created an accessible data source.

Context

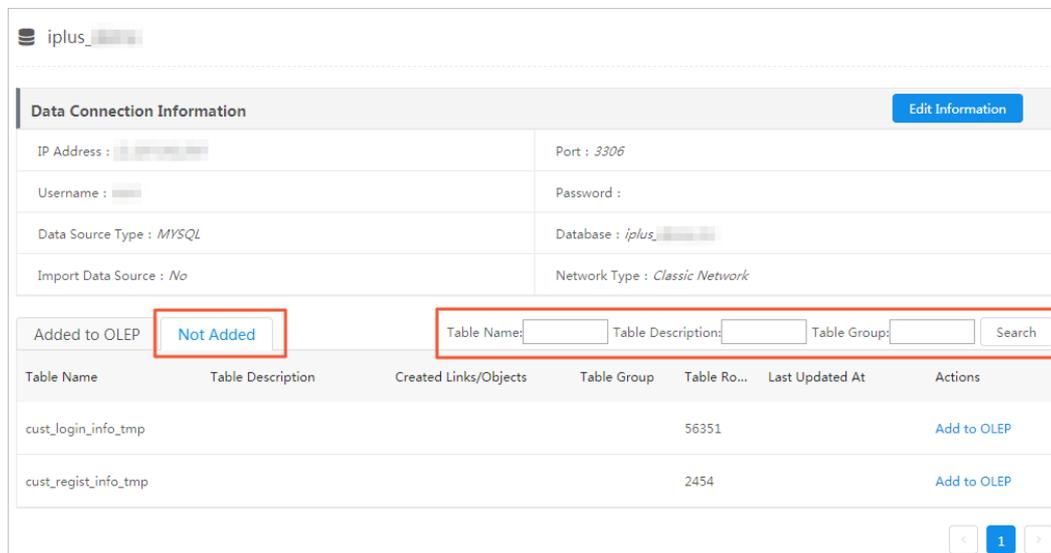
OLEP models include the following types of mappings: table-to-object mappings, table-to-link mappings, and table-to-event mappings. You can add objects, links, and events when you create OLEP models. After that, you can view and configure these objects, links, and events on the **Object Information**, **Link Information**, and **Event Information** pages, respectively. You can configure these items to reflect your business semantics. An OLEP table serves as a source for configuring objects, first-degree links, and events. A table can be mapped to multiple objects, links, and events.

Procedure

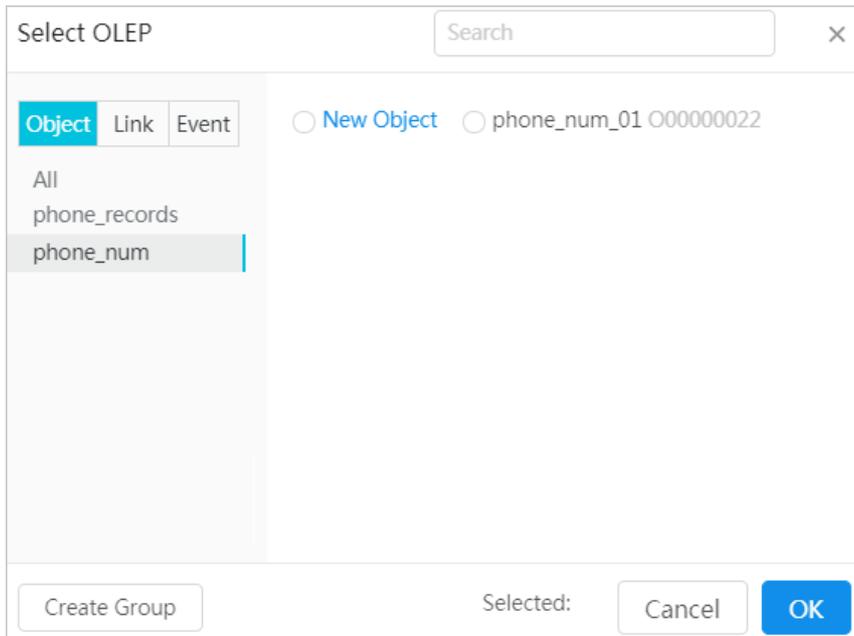
1. [Log on to Administration Console of Graph Analytics.](#)
2. Click **Source Tables** in the top navigation bar. The **Data Sources** page appears.
3. Click a data source in the left-side navigation pane. The data source details are displayed on the right side of the page.
4. Click the **Not Added** tab. All tables that have no OLEP models are displayed.

You can search for a table quickly and accurately by specifying the Table Name, Table Description, or Table Group parameter.

Tables that have no OLEP models



5. Select a table, and then click **Add to OLEP** in the Actions column. The **Select OLEP** dialog box appears.



The **Select OLEP** dialog box contains the **Object**, **Link**, and **Event** tabs which are used to create mappings to objects, links, and events, respectively. For more information about how to create a mapping to an object, link, or event, see [Step 6](#), [Step 7](#), and [Step 8](#).

If no existing object groups, link groups, or event groups meet your requirements, click **Create Group** to add an object group, link group, or event group.

6. Map the table to an object.

- i. Select **New Object** or an existing object and then click **OK**. The **Map to Object** dialog box appears.

Map to a newly created object

Map to Object

* Object Name: Group:

Property ID	Table Column	* Property Name	* Primary Key	Mapping <input type="checkbox"/>
O00000027P0001	<input type="text" value="identity_card"/>	<input type="text" value="identity_card"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
O00000027P0002	<input type="text" value="name"/>	<input type="text" value="name"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
O00000027P0003	<input type="text" value="phone_num"/>	<input type="text" value="phone_num"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

< 1 >

Cancel OK

Map to an existing object

Map to Object

Object Name: Group:

Property ID	Table Column	Property Name	Primary Key
O00000022P0001	<input type="text" value="identity_card"/>	<input type="text" value="identity_card"/>	<input type="checkbox"/>
O00000022P0002	<input type="text" value="name"/>	<input type="text" value="name"/>	<input type="checkbox"/>
O00000022P0003	<input type="text" value="phone_num"/>	<input type="text" value="phone_num"/>	<input checked="" type="checkbox"/>

< 1 >

Cancel OK

ii. Configure the parameters as needed.

- **New object:** Configure the parameters based on [Parameters used to map the table to a new object](#).
- **Existing object:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is greater than the number of existing properties of the object, you can click **Add Property** to add new properties.

Parameters used to map the table to a new object

Field	Description
Object Name	The user-defined object name. It must be unique.
Group	The object group to which the object belongs. All available object groups are displayed in the drop-down list.
Property Name	The name of an object property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, property names are displayed instead of the actual table columns that are mapped to the properties.
Mapping	Indicates whether to enable the property mapping.
Primary Key	The property that is used as a primary key. Each primary key uniquely identifies an object. You must configure one or more properties as primary keys for each object. You must enable Mapping for primary keys.

iii. Click **OK**.

7. Map the table to a link.

- i. Click the **Link** tab. All first-degree links to which the current table has been mapped are displayed.
- ii. Select **New Link** or an existing link and then click **OK**. The **Map to Link** dialog box appears.

Map to a newly created link

Map to Link ✕

* Link Name: Group:

* Source Object: * Target Object:

Basic Information

Property ID	Table Column	* Property Name	Mapping <input checked="" type="checkbox"/>
L00000016P0001	<input type="text" value="callee_num"/>	<input type="text" value="callee_num"/>	<input checked="" type="checkbox"/>
L00000016P0002	<input type="text" value="caller_num"/>	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>

< 1 >

^ Source Property Mapping

SourceObject Property:phone_num_01 - phone_num * Link Property:

^ Target Property Mapping

TargetObject Property:phone_num_01 - phone_num * Link Property:

Map to an existing link

Map to Link ✕

Link Name: Group:

Source Object: Target Object: Create Property

Basic Information

Property ID	Table Column	Property Name
L00000014P0001	<input type="text" value="caller_num"/> *	<input type="text" value="caller_num"/>
L00000014P0002	<input type="text" value="callee_num"/> *	<input type="text" value="callee_num"/>

< 1 >

^ Source Property Mapping

SourceObject Property:phone_num_01 - phone_num * Link Property:

^ Target Property Mapping

TargetObject Property:phone_num_01 - phone_num * Link Property:

iii. Configure the parameters as needed.

- **New link:** Configure the parameters based on [Parameters used to map the table to a new link](#).
- **Existing link:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is more than the number of existing link properties, you can click **Add Property** to add new link properties.

Parameters used to map the table to a new link

Field	Description
Link Name	The user-defined link name. It must be unique.
Group	The link group to which the link belongs. All available link groups are displayed in the drop-down list.
Source	The source object of the link. You can select an object from the drop-down list. The Source Property Mapping parameter is displayed only after you configure the Source Object parameter.
Target	The target object of the link. You can select an object from the drop-down list. The Target Property Mapping parameter is displayed only after you configured the Target Object parameter.
Property Name	The name of a link property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, the Property Name values are displayed as the table header in Details on the Graph page.
Switch	Indicates whether to enable the property mapping.
Link Property in Source Property Mapping	The link property to which a primary key property of the source object is mapped.
Link Property in Target Property Mapping	The link property to which a primary key property of the target object is mapped.

iv. Click **OK**.

8. Map the table to an event.

- i. Click the **Event** tab. All events to which the current table has been mapped are displayed.
- ii. Select **New Event** or an existing event and then click **OK**. The **Map to Event** dialog box appears.
 Map to a newly created event

Map to Event ✕

* Event Name : Group:

Basic Information

Property ID	Table Column	* Property Name	* Primary Key	Mapping <input type="checkbox"/>
E00000014P0001	<input type="text" value="callee_num"/>	<input type="text" value="callee_num"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
E00000014P0002	<input type="text" value="caller_num"/>	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

< 1 >

Primary Key Mappings of Correlated Objects Add Mapping

<input type="text" value="phone_num O00000022"/>	phone_num(O00000022P0003) :	<input type="text" value="caller_num E00000014P00..."/>
<input type="text" value="phone_num O00000022"/>	phone_num(O00000022P0003) :	<input type="text" value="callee_num E00000014P00..."/>

Map to an existing event

Map to Event ✕

Event Name : Group: Add Property

Basic Information

Property ID	Physical Table Field	Property Name	Primary Key
E00000014P0001	<input type="text" value="caller_num"/> *	<input type="text" value="callee_num"/>	<input type="checkbox"/>
E00000014P0002	<input type="text" value="callee_num"/> *	<input type="text" value="caller_num"/>	<input checked="" type="checkbox"/>

< 1 >

^ Primary Key Mappings of Correlated Objects

phone_num(O00000022P0003) :

phone_num(O00000022P0003) :

OK

iii. Configure parameters as needed.

- **New event:** Configure the parameters based on [Parameters used to map the table to a new event](#).
- **Existing event:** Create a one-to-one mapping between each **Table Column** and **Property Name** as needed.

If the number of table columns is greater than the number of existing event properties, you can click **Add Property** to add new event properties.

Parameters used to map the table to a new event

Field	Description
Event Definition Name	The user-defined event name. It must be unique.
Group	The event group to which the event belongs. All available event groups are displayed in the drop-down list.
Property Name	The name of an event property to which a table column is mapped. By default, the property name is the same as the column name. You can also define the property name as needed. On the Analytics Workbench, the Property Name values are displayed as the table header in Details on the Graph page.
Switch	Indicates whether to enable the property mapping.
Primary Key	The property that is used as a primary key. Each primary key uniquely identifies an event. You must configure one or more properties as primary keys for each event. Switch must be turned on for the properties that are configured as primary keys.
Map Primary Keys to Correlated Objects	The mappings between the primary keys of correlated objects and the event properties. Two or more correlated objects are required. You can click Add Mapping to add more necessary mappings between the primary keys of correlated objects and the event properties. You must enable Mapping for the event properties to which the primary keys of the correlated objects are mapped.

iv. Click **OK**.

9. After you create OLEP models for the data tables, click the **Added to OLEP** tab to check the results.

8.3.2.2. View an OLEP table

After you have created OLEP models for a table, you can view the created OLEP table.

Prerequisites

OLEP models have been created for a table.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Select the target table you want to view from the left-side data source list.

The table information is displayed in the right-side pane, including table basics, dependent objects and links, and table columns, as shown in [View tables](#).

View tables

cust_regist_info_demo

Table Information Edit Remove

Table Name : *cust_regist_info_demo* Table Description : Table Size : Table Rows :

Created Objects : *device,cust_ip*

Created Links : *cust_device,cust_ip*

Created Events :

Created Dictionaries :

Added Columns Columns Not Added Column Name : Column Description : Search

All [device](#) [cust](#) [ip](#) [cust_device](#) [cust_ip](#)

Column Name	Column Description	Property Name	Property ID	Column Type	String Length	Primary Key	Timestamp Format	Actions
ip_prov	--	--	--	string		No	None	Map
mac		mac	O00000156P0005	string		No	None	Remove
date_time	--	--	--	time		No	yyyy-MM-dd HH:mm:ss	Map

8.3.2.3. Edit OLEP tables

After you have created OLEP models for a table, you can still add new object, link, or event mappings to the table. You can also add a description for the table.

Prerequisites

Make sure that OLEP models have been created for a table.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click a table in the left-side navigation pane. The data source details are displayed on the right side of the page.
Add new object, link, or event mappings to tables.
3. On the **Added to OLEP** tab, select a table, and then click **Add Mapping**. The **Select OLEP** dialog box appears. You can add a new object, link, or event mapping to the table. For more information, see the procedure described in [Create OLEP models for tables.](#)

Modify the table description.

4. You can use one of the following methods to modify the table description.

Method	Operation
On the Added to OLEP page	<ol style="list-style-type: none"> i. Click the Edit icon  next to the Table Description column. You can then modify the Table Description. ii. Click OK to save the changes. A success message is displayed after the operation is completed.

Method	Operation
On the table details page	<ul style="list-style-type: none"> i. Click a table in the left-side navigation pane. On the right side of the page, click Edit in the Table Information section. You can then modify the Table Description. ii. Click OK to save the changes. A success message is displayed after the operation is completed.

8.3.2.4. Remove an OLEP table

If the mapping between a table and an object, link, or event is no longer used, you can remove the mapping.

Prerequisites

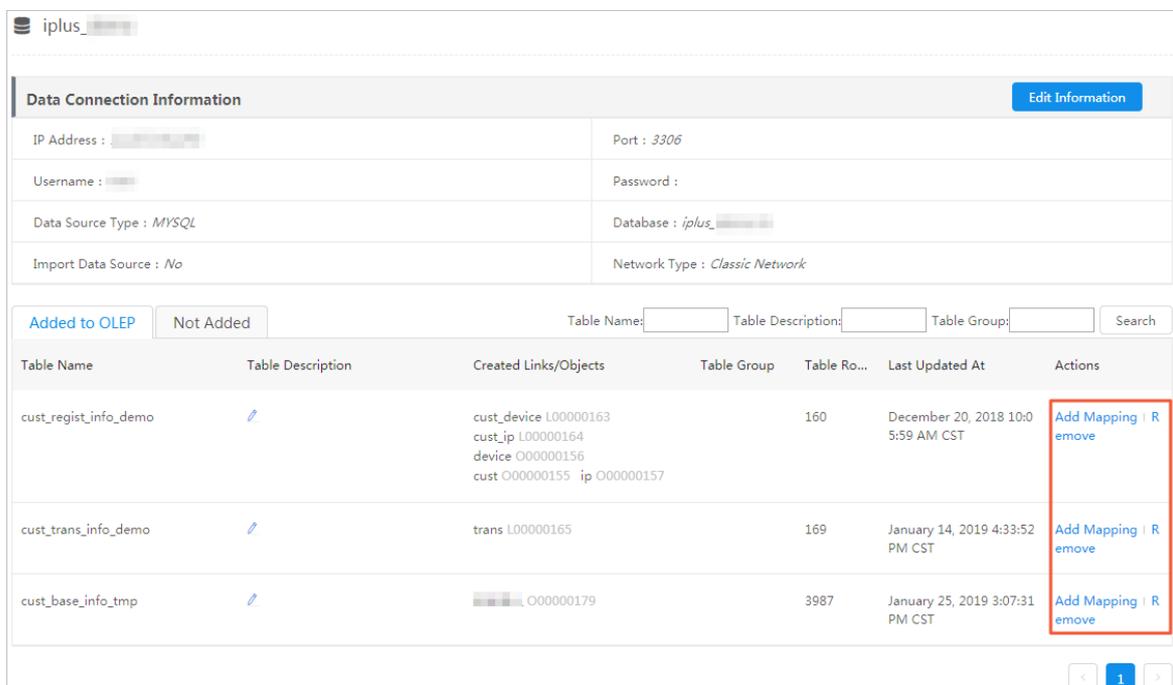
You can access the data source that stores the table.

Context

You only remove the mappings between the tables and objects, links, and events. The objects, links, events, and tables will not be deleted.

Procedure

1. Click the **Source Tables** tab in the top navigation bar. The **Data Sources** page appears.
2. Click a table in the left-side navigation pane. The data source details are displayed in the right-side pane.
3. On the **Added to OLEP** tab, click **Remove** next to a table.



4. In the **Select OLEP** dialog box, select an **Object**, **Link**, or **Event** mapping. You can select only one mapping at a time.

5. Click **OK**.

If all mappings are removed from a table, the table will be automatically moved from the **Added to OLEP** tab to the **Not Added** tab.

8.3.3. OLEP table columns

8.3.3.1. Add OLEP table columns

If a data table has been mapped to an object, link, or event, and the table still has unoccupied columns (columns that are not correlated with any object, link, or event), you can add these columns to the existing mappings as needed.

Prerequisites

A data table has been mapped to an object, link, or event, but the table still has unoccupied columns.

Context

Before you configure the OLEP table columns, sort out the columns for which you will create OLEP models and data types of the columns, especially the time columns.

Procedure

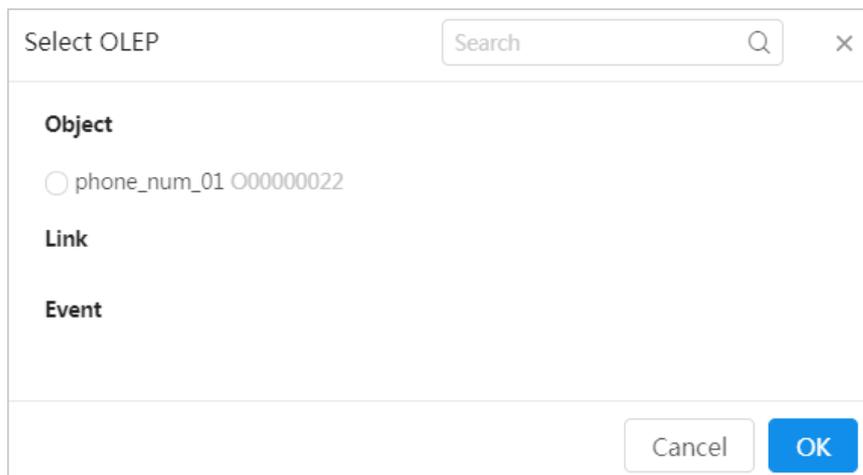
1. [Log on to Administration Console of Graph Analytics.](#)
2. Click **Source Tables** in the top navigation bar.
3. On the **Data Sources** page, click a data source in the left-side navigation pane, and then click the table to which you want to add the columns.
4. Click the **Columns Not Added** tab in the right-side area. In the **Columns Not Added** tab that appears, click **Add** in the Actions column.

Columns displayed in the **Columns Not Added** tab are not mapped to any object, link, or event.

5. In the **Select OLEP** dialog box that appears, select the object, link, or event to which the columns map, and then click **OK**.

The **Select OLEP** tab only displays the objects, links, and events that have mapped to the current data table.

The following example demonstrates how to add columns to an object.



6. In the **Map to Object** dialog box that appears, select the object properties that map to the columns you want to add.

Object properties that have been mapped to the current data table are dimmed and cannot be operated.

Property ID	Property Name	Map
O00000022P0001	identity_card	<input type="checkbox"/>
O00000022P0002	name	<input checked="" type="checkbox"/>
O00000022P0003	phone_num	<input type="checkbox"/>

< 1 >

Cancel OK

7. After you configure the preceding parameters, click **OK**.

8.3.3.2. Edit OLEP table columns

After a data table is mapped to objects, links, or events, you can modify the **Column Description** and **Timestamp Format** of added columns.

Prerequisites

The data table has been mapped to an object, link, or event.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click **Source Tables** in the top navigation bar.
3. On the **Data Sources** page, click a data source in the left-side navigation pane, and then click the table that contains the columns you want to modify.

After you select the table, the details of the table will be displayed in the right-side **Data Sources** pane.

4. Click the **Added Columns** tab.

By default, the **Added Columns** tab displays all columns that have been added.

cust_regist_info_demo

Table Information Edit Remove

Table Name : cust_regist_info_demo Table Description : Table Size : Table Rows :

Created Objects : device,cust,ip

Created Links : cust_device,cust_ip

Created Events :

Created Dictionaries :

Added Columns Columns Not Added Column Name : Column Description : Search

All device cust ip cust_device cust_ip

Column Name	Column Description	Property Name	Property ID	Column Type	String Length	Primary Key	Timestamp Format	Actions
ip_prov		ip_prov	O00000157P0004	string		No	None	Edit
mac		mac mac	O00000156P0005 L00000163P0005	string		No	None	Edit
date_time		date_time date_time	L00000163P0002 L00000164P0002	time		No	yyyy-MM-dd HH:mm:ss	Edit
ip		ip ip	O00000157P0003 L00000164P0003	string		No	None	Edit

- On the **All** tab, click **Edit** in the **Actions** column to modify the **Column Description** and **Timestamp Format** of a column.

Added Columns Columns Not Added Column Name : Column Description : Search

All device cust ip cust_device cust_ip

Column Name	Column Description	Property Name	Property ID	Column Type	String Length	Primary Key	Timestamp Format	Actions
ip_prov	<input type="text"/>	ip_prov	O00000157P0004	string		No	None	Save
mac		mac mac	O00000156P0005 L00000163P0005	string		No	None	Edit
date_time		date_time date_time	L00000163P0002 L00000164P0002	time		No	yyyy-MM-dd HH:mm:ss	Edit
ip		ip ip	O00000157P0003 L00000164P0003	string		No	yyyy-mm-dd	Edit
cust_id	cust_id	cust_id cust_id cust_id	O00000155P0001 L00000163P0001 L00000164P0001	string		No	yyyy/mm/dd	Edit

The **Timestamp** option in the **Timestamp Format** drop-down list refers to the UNIX timestamp.

- After you have configured the preceding parameters, click **Save**.

8.3.3.3. Remove OLEP table columns

After you have created OLEP models for a table, you can remove unnecessary OLEP mappings of specific columns in the table.

Prerequisites

OLEP table columns are not referenced.

Context

After a table has been mapped to objects, links, or events, you cannot separately remove the following mappings of a column:

- Table-to-object mappings: The mapping between the column and a primary key property of the object cannot be removed.
- Table-to-link mappings: The mapping between the column and a correlated property of the link cannot be removed.
- Table-to-event mappings: The mapping between the column and a primary key property of the event cannot be removed.

To remove these mappings, remove the OLEP mapping from the table and then add OLEP mappings to the table again. For more information, see [Remove OLEP tables](#) and [Create OLEP models for tables](#).

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click **Source Tables** in the top navigation bar.
3. On the **Data Sources** page, click a data source in the left-side navigation pane, and then click the table that contains the columns you want to remove.
4. On the right side of the **Data Sources** page, click the **Added Columns** tab.
By default, the **Added Columns** tab displays all columns that have been added.
5. On the **Added Columns** tab, click the tab of an object, link, or event that has been mapped to a data table.

On the selected object, link, or event page, the table columns that have been mapped are highlighted.

The screenshot shows the configuration page for the table 'cust_regist_info_demo'. It includes a 'Table Information' section with fields for Table Name, Description, Size, and Rows. Below this is a section for 'Added Columns' with a search bar. A list of tabs is shown, with 'device' selected and circled in red. A table lists columns with their properties, and the 'mac' row is highlighted in light blue with its 'Remove' button circled in red.

Column Name	Column Description	Property Name	Property ID	Column Type	String Length	Primary Key	Timestamp Format	Actions
ip_prov		--	--	string		No	None	Map
mac		mac	O00000156P0005	string		No	None	Remove
date_time		--	--	time		No	yyyy-MM-dd HH:mm:ss	Map

6. Select a column, and then click **Remove** to remove the object, link, or event mapping.

If all mappings of a column are removed, including object, link, and event mappings, the column will be automatically moved to the **Columns Not Added** tab.

8.4. Dictionaries

8.4.1. Create a dictionary

This topic describes how to create a dictionary for the columns in a data table of a data source to establish a mapping between a key column and a value column in the data table. This operation helps you convert a column name that is not easy to understand into a column name that is easy to understand.

Prerequisites

- A data source is created. For more information, see [Create data sources](#).
- The mappings between tables and objects, links, or events are configured. For more information, see [Create OLEP models for tables](#).
- OLEP table columns are added. For more information, see [Add OLEP table columns](#).

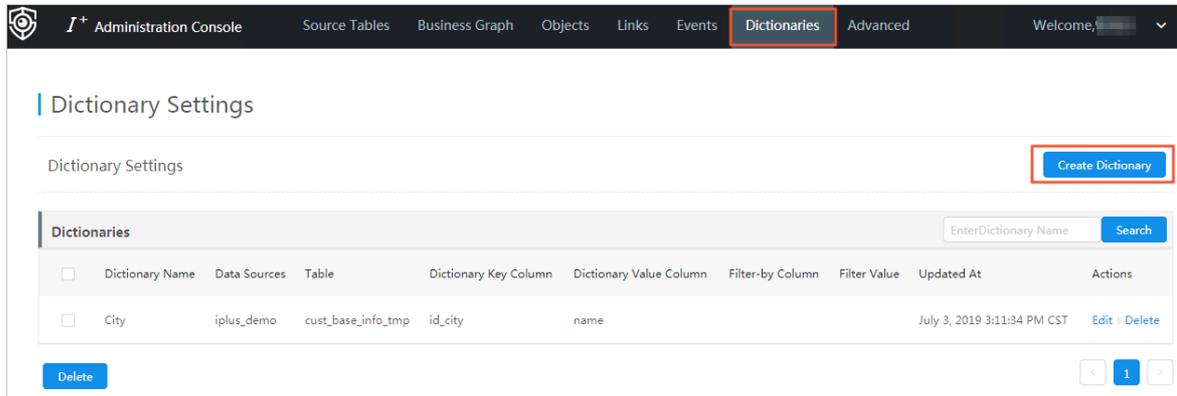
Context

A dictionary is used to map a specific key column in a table to a value column and display the value column name in Analytics Workbench instead of the original column name.

For example, on the **Graph** page of Analytics Workbench, column A that represents **company code** is displayed. After column A is escaped to column B that represents **company name**, the company name that corresponds to the company code is displayed on the **Graph** page.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, choose **Dictionaries**.



3. On the **Dictionary Settings** page, click **Create Dictionary** in the upper-right corner of the page.

The screenshot shows a 'Dictionary Settings' dialog box with the following fields:

- * Dictionary Name:** Text input field with placeholder 'Enter a dictionary name.'
- * Data Sources:** Dropdown menu.
- * Table:** Dropdown menu.
- * Key Column:** Dropdown menu with a help icon (?)
- * Dictionary Value:** Dropdown menu with a help icon (?)
- Filter-by Column:** Dropdown menu with a help icon (?)
- Filter Operator:** Dropdown menu.
- Filter Value:** Text input field with placeholder 'Enter a filter value.' and a help icon (?)

Buttons: Cancel, OK

4. In the **Dictionary Settings** dialog box that appears, specify the parameters as needed.

The following table describes these parameters.

Parameter configurations for creating a dictionary

Parameter	Description
Dictionary Name	The name of the dictionary. You can customize the name as needed.
Data Sources	The data sources that you want to reference.
Table	The table in the data source that you want to reference.
Key Column	The column that stores the dictionary code in the selected data table.
Dictionary Value Column	The column that explains its corresponding key column in the selected data table.
Filter-by Column	Optional parameters. They are used to filter dictionary tables based on different conditions. If you specify any of the three parameters, the other two parameters are required.
Filter Operator	
Filter Value	

5. After you complete the preceding parameter configurations, click **OK**.

What to do next

After you create a dictionary for a column in a data table, when you map a property of an object, link, or event to a column in the data table, select **Dictionary** for **Display Type** for the property. For more information, see [Configure object properties and business parameters](#), [Configure link properties and business parameters](#), or [Configure event property parameters](#).

8.4.2. Modify a dictionary

This topic describes how to modify a dictionary in a dictionary list.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Dictionaries**. The Dictionary Settings page appears. All existing dictionaries are displayed.
3. Select a dictionary and then click **Edit**, as shown in [Modify a dictionary](#).

Modify a dictionary

The screenshot shows a 'Dictionary Settings' dialog box with the following fields and values:

- * Dictionary Name: City
- * Data Sources: iplus_demo
- * Table: cust_base_info_tmp
- * Key Column: id_city
- * Dictionary Value: name
- Filter-by Column: (empty)
- Filter Operator: (empty)
- Filter Value: Enter a filter value.

Buttons: Cancel, OK

4. In the dialog box that appears, modify the parameters as needed. For more information about dictionary parameters, see [Parameter description](#).
5. Click **OK**.

8.4.3. Delete a dictionary

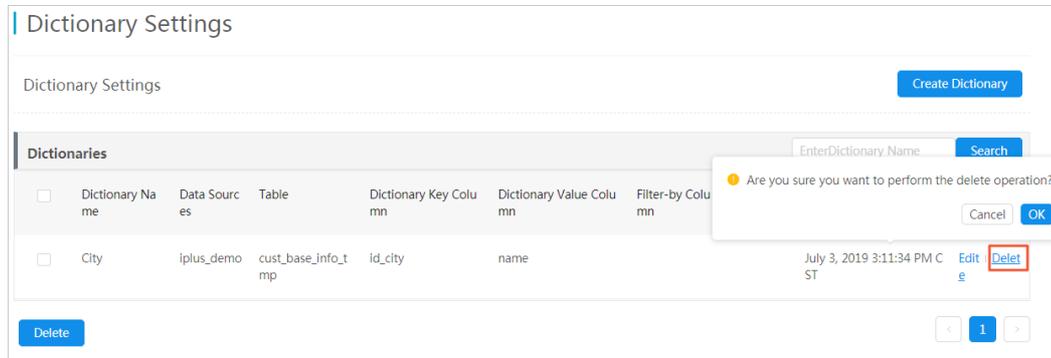
This topic describes how to delete a dictionary. You cannot delete dictionaries that have been referenced.

Delete a dictionary

1. [Log on to Administration Console of Graph Analytics](#).

- In the top navigation bar, click **Dictionary**s. The Dictionary Settings page appears. All existing dictionaries are displayed.
- Select a dictionary, and then click **Delete**. A confirm message appears, as shown in [Delete a dictionary](#).

Delete a dictionary

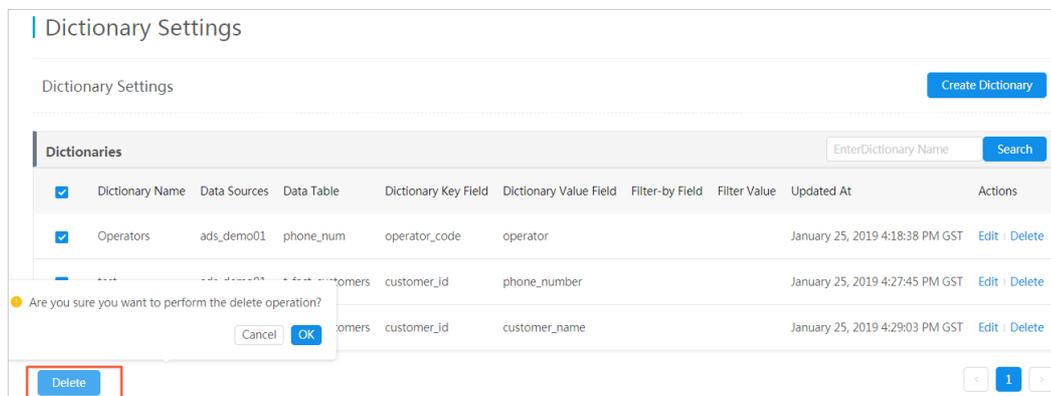


- Click **OK**.

Delete multiple dictionaries at a time

- [Log on to Administration Console of Graph Analytics](#).
- In the top navigation bar, click **Dictionary**s. The Dictionary Settings page appears. All existing dictionaries are displayed.
- Select one or more dictionaries, and then click **Delete** in the lower-left corner. A confirm message appears, as shown in [Delete multiple dictionaries at a time](#).

Delete multiple dictionaries at a time



- Click **OK**.

8.5. Object information

8.5.1. Object groups

8.5.1.1. Create an object group

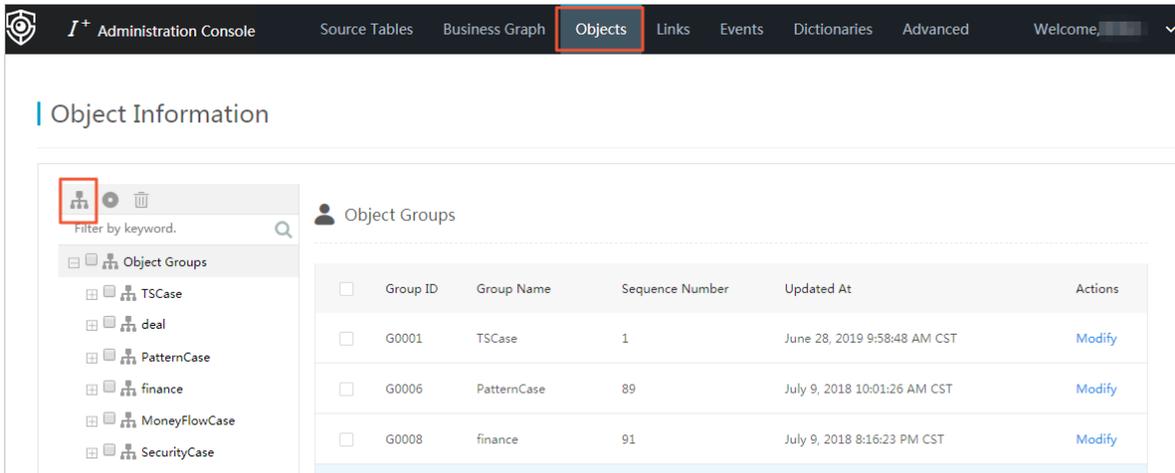
You can use object groups to classify objects, so that you can search for and manage objects with ease. An object must be and can only be included in one object group. You need to create a proper object group before you create an object.

Prerequisites

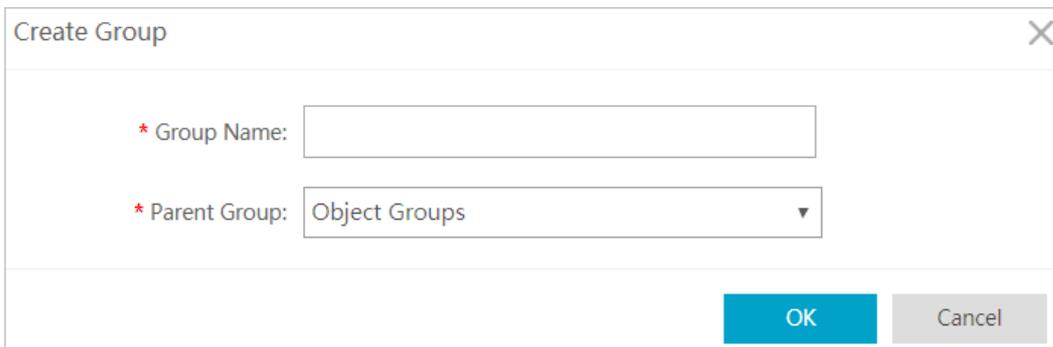
You have an account and password for Graph Analytics and have permissions to perform operations in Administration Console of Graph Analytics.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Objects**.
3. Click the **Create Group** icon  to create a group.



4. In the **Create Group** dialog box that appears, specify **Group Name** and **Parent Group**.



5. Click **OK**.

8.5.1.2. View object groups and objects

Graph Analytics allows you to view all object groups in the current environment. This function makes it easy for you to understand the existing object groups and the object information under each group at any time.

Prerequisites

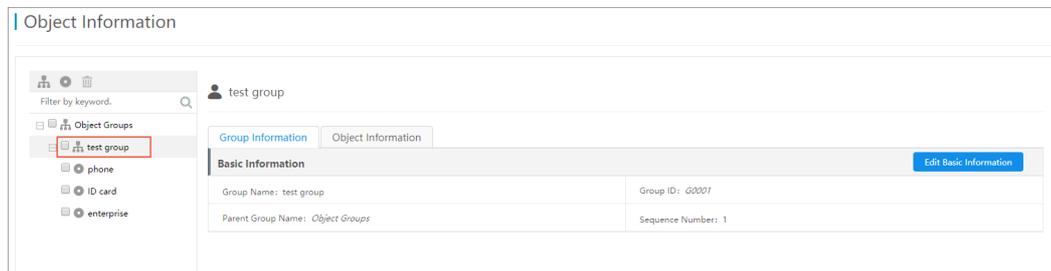
You have an account and password for Graph Analytics and have permissions to perform operations in Administration Console of Graph Analytics.

Procedure

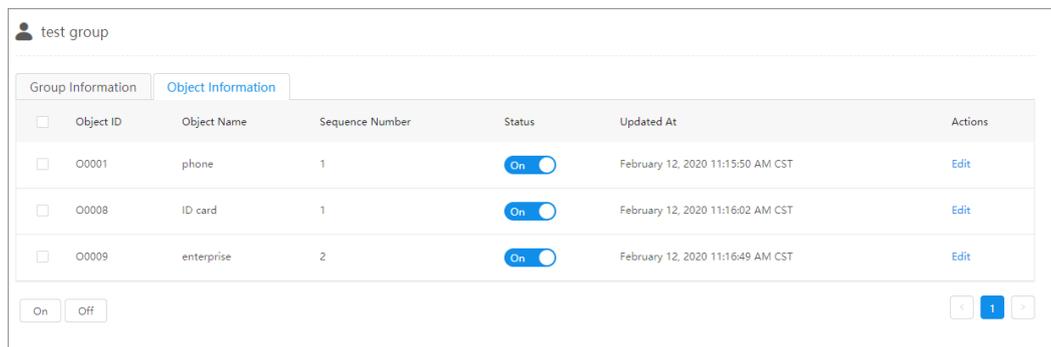
1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane of the **Object Information** page, click an object group. The **Group Information** and **Object Information** tabs of the object group are displayed in the right-side pane.

The left-side navigation pane displays all the object groups in the current environment. You can view the groups one by one.

Group Information



Object Information



4. Click OK.

8.5.1.3. Modify object groups and objects

In Graph Analytics, you can modify the name and sequence number of an object or object group. You can adjust the basic information of an object or object group at any time if required.

Prerequisites

- You have created an object group. For more information about how to create an object group, see [Create an object group](#).
- You have created an object that belongs to this object group. For more information about how to create an object, see [Create an object](#).
- You have deleted the dependency information of an object, including the mapping between the object and a data table if you need to disable this object.

Context

This topic describes the following operations:

- Modify the basic information about an object group
- Modify the basic information about an object
- Disable and enable an object

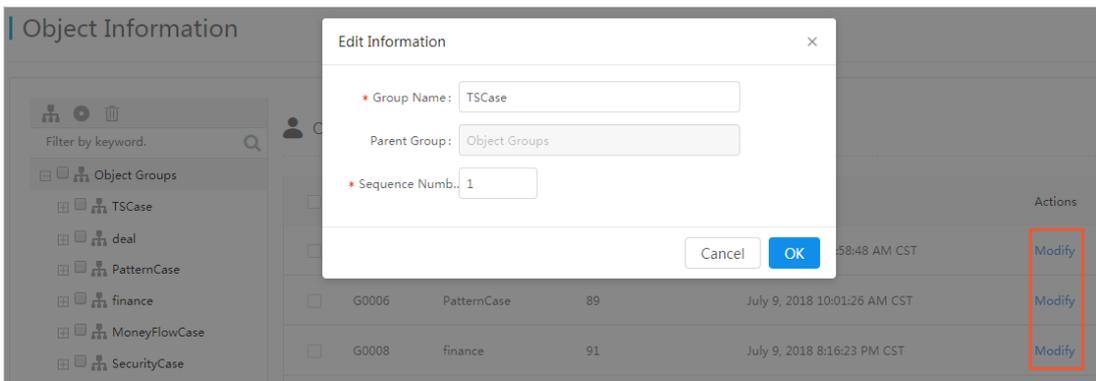
After you have created an object by configuring the object properties, the object is automatically enabled. You can disable an object if it is no longer used for a period of time. You can also enable it again when necessary. You cannot use an object in Analytics Workbench after the object is disabled.

Modify the basic information about an object group

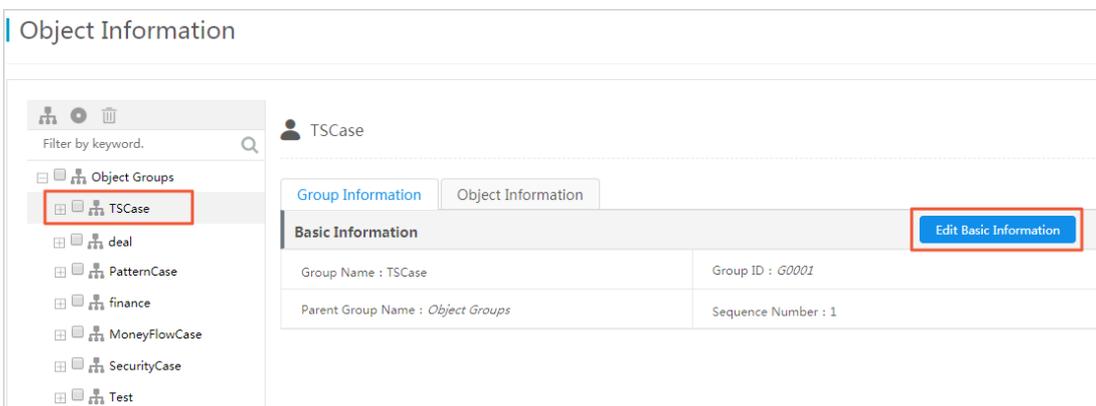
You can modify the basic information about an object group by using one of the following methods:

Method	Procedure
Method 1	<ol style="list-style-type: none"> 1. Log on to Administration Console of Graph Analytics. 2. In the top navigation bar, click Objects. 3. In the Object Groups area, select an object group and click Modify in the Actions column. 4. In the Edit Information dialog box that appears, specify Group Name and Sequence Number, as shown in Edit information. 5. Click OK.
Method 2	<ol style="list-style-type: none"> 1. Log on to Administration Console of Graph Analytics. 2. In the top navigation bar, click Objects. 3. In the left-side navigation pane, click the object group you want to modify. 4. On the right-side Group Information tab, click Edit Basic Information. Specify Group Name and Sequence Number, as shown in Edit information. 5. Click Save.

Edit information



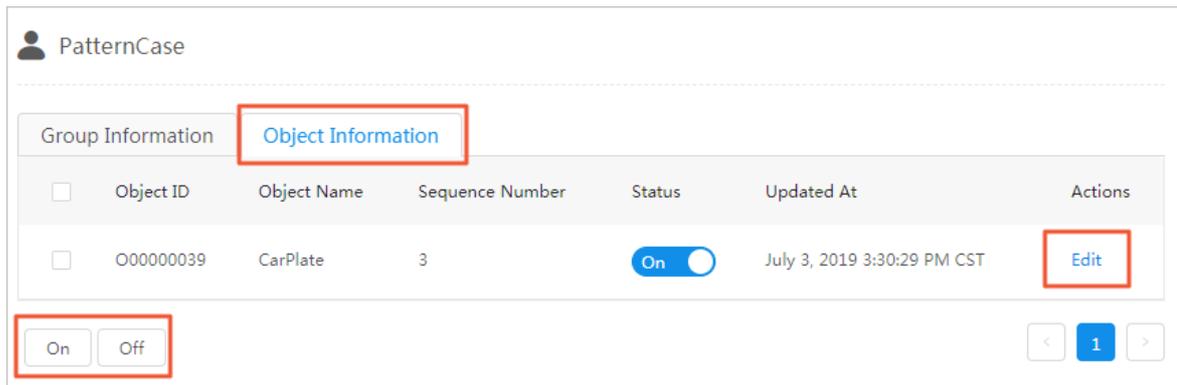
Edit basic information



Modify basic object information

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane, click the object group to which the object belongs, and then click the **Object Information** tab on the right side of the page.

- On the **Object Information** tab, select an object and then click **Edit** in the Actions column.



- In the **Edit Information** dialog box that appears, reconfigure **Object Name** and **Sequence Number**.
- Click **OK**.

Disable and enable an object

- Log on to [Administration Console of Graph Analytics](#).
- In the top navigation bar, click **Objects**.
- In the left-side navigation pane, click the object group to which the object belongs, and then click the **Object Information** tab on the right side of the page.
- You can use the following methods to enable or disable an object on the **Object Information** tab:

Method	Procedure
Disable an object	<p>Disable an object: Click the  icon next to the object you want to disable.</p> <p>Disable multiple objects: Select the objects you want to disable and then click Off in the lower part of the page.</p> <p>After an object is disabled, the Status changes from a highlighted  icon to a dimmed  icon.</p>
Enable an object	<p>Enable an object: Click the  icon next to the object you want to enable.</p> <p>Enable multiple objects: Select the objects you want to enable and then click On.</p> <p>After the object is enabled, the Status changes from a dimmed  icon to a highlighted  icon.</p>

8.5.1.4. Delete object groups and objects

If some objects or object groups are no longer used, you can delete these objects and object groups.

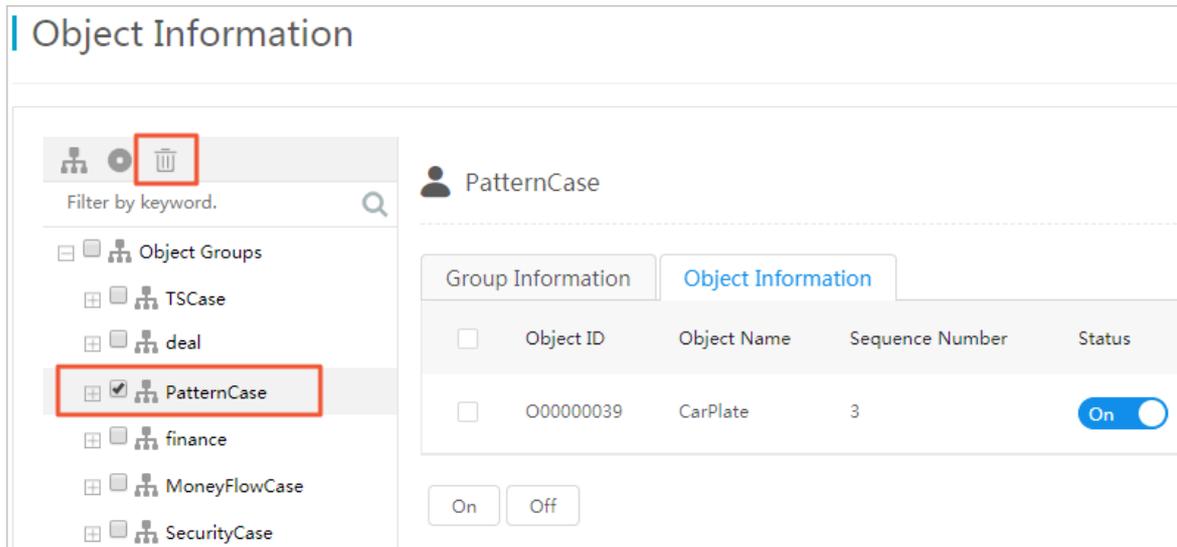
Prerequisites

- The dependency information of an object, for example, the mapping between the object and the physical table, has been deleted before you delete the object.
- All the objects in an object group have been deleted before you delete the object group. Only empty

object groups can be deleted.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Objects**.
3. (Optional) If there are still objects in the object group, you must delete these objects first.
 - i. In the left-side navigation pane, select all objects in the object group you want to delete, and click the  icon in the upper-left corner.
 - ii. In the **Delete Object Information** dialog box that appears, click **OK** to clear the object group.
4. In the left-side navigation pane, select the object group you want to delete and then click the  icon.



5. In the **Delete Object Information** dialog box that appears, click **OK**.

8.5.2. Objects

8.5.2.1. Create an object

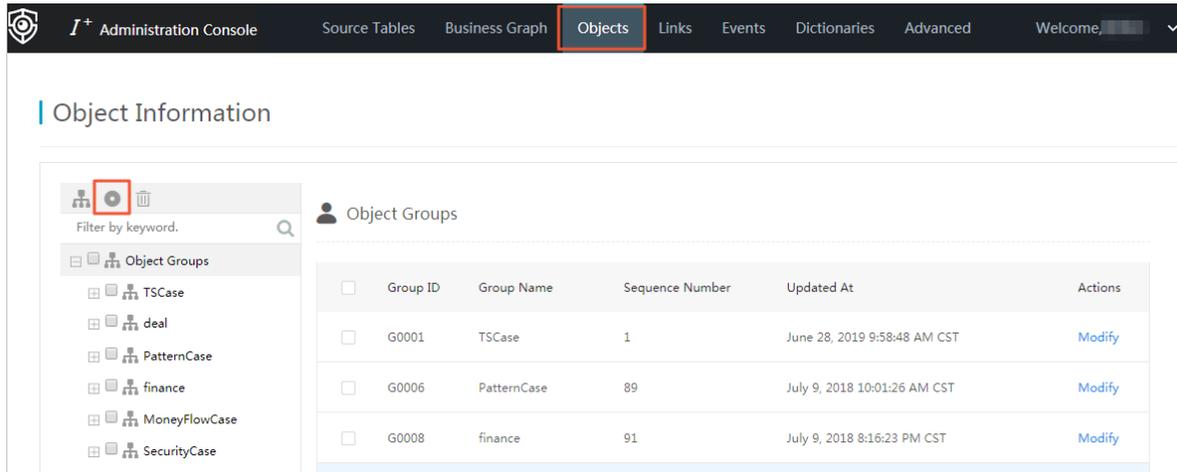
In Graph Analytics, objects are mapped to entities in the real world. Before you perform a relationship analysis on an entity, you must create an object that corresponds to the entity based on the data you have obtained. A complete object contains the basic information, property information, and relevant parameters. This topic describes how to configure the basic information of an object.

Prerequisites

An object group has been created. For more information, see [Create an object group](#).

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Objects**.



3. On the **Object Information** page, click the **Create Object** icon.
4. Configure the object information in the **Create Object** dialog box that appears.

The parameters are described in [Parameters of created objects](#).

Create Object ✕

* Object Name:
(The object name must be 1 to 16 characters in length and can contain letters, numbers, and Chinese characters.)

Description:

* Group:

* Add in Graph:

* Object Icon
 Display Position:

* Allow Table
 Mapping:

* Object Icon: Icon Library Referenced URL


 Enterprise3


 File_Folder


 Enterprise_A...


 Private_Acc...


 Person


 Enterprise


 Enterprise1


 ip

Parameters of created objects

Parameter	Description
Object name	The user-defined object name. It must be unique.
Object Description	The object description that helps you understand the object.
Group	By default, the selected object group in the left-side navigation pane is used. You can also select another group as needed.
Add in Graph	Indicates whether you are allowed to manually add an object node to the graph page.
Object Icon Display Position	Valid values: <ul style="list-style-type: none">○ Use Icon in Graph, Not Show Image on Right-side Pane○ Use Icon in Graph, Show Image on Right-side Pane○ Use Image in Graph, Not Show Image on Right-side Pane○ Use Image in Graph, Show Image on Right-side Pane
Allow Table Mapping	By default, this parameter is set to Yes.
Object Icon	Allows you to set the icon of the object. You can select an icon in Icon Library or enter a URL to reference an external icon.

5. Click **OK**.

What's next

1. After you have created an object, you must configure the properties and business parameters based on your business requirements. For more information, see [Configure object properties and business parameters](#).
2. After you have configured the properties and business parameters of an object, you must log on to Analytics Workbench again to use the new object.

8.5.2.2. Configure object properties and business parameters

After you add an object, you need to configure the business parameters of the object based on your requirements so that you can view and analyze the object in Analytics Workbench.

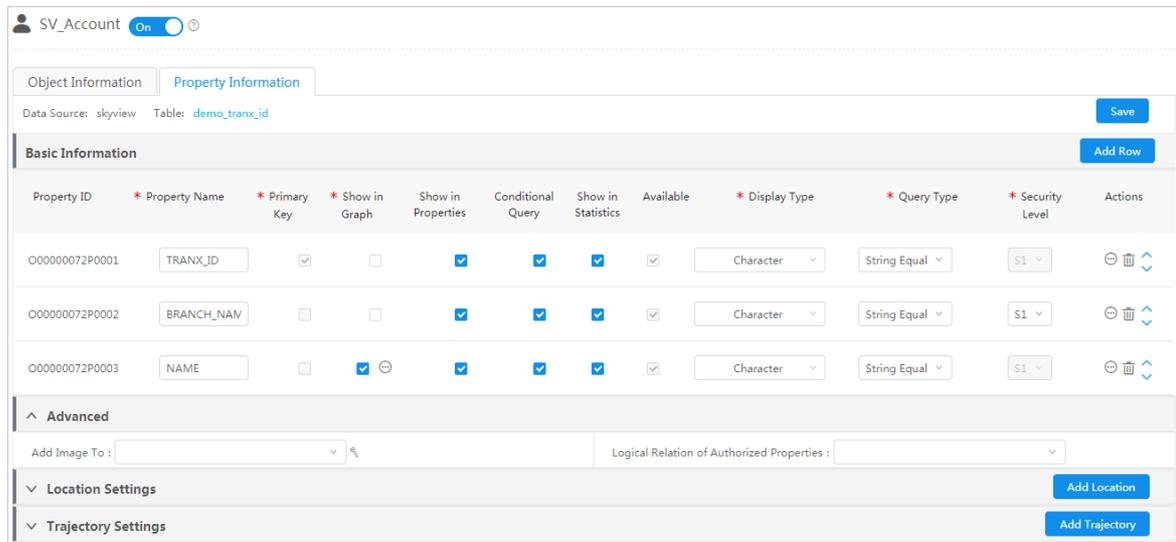
Prerequisites

- You have created a data source. For more information, see [Create data sources](#).
- You have configured mappings between tables and objects, links, or events. For more information, see [Create OLEP models for tables](#).

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click **Objects** on the top of the page.
3. In the left-side navigation pane of the **Object Information** page, click the name of the object you want to configure and then click the **Property Information** tab on the right side.

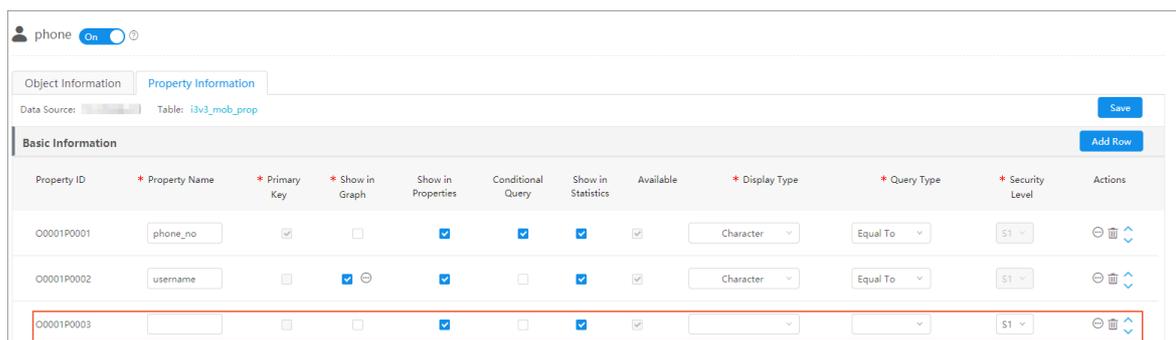
If this object has been mapped to a physical table in the data source, the **Property Information** tab displays the **Data Source** and **Table** information.



4. Specify parameters in the **Basic Information** section.

Click **Add Row** to add a property in **Basic information**.

Parameters in Basic Information describes the parameters in **Basic Information**.



Parameters in Basic Information

Parameter	Description
Property ID	The ID of a property. It is automatically generated.
Property Name	The property name that is displayed on Analytics Workbench. Enter a name that describes your business.
Primary key	<p>You must select one or more primary keys for the properties when you add an object for the first time. After the configuration is complete, you cannot modify or delete the primary keys.</p> <p>If you add a node in Analytics Workbench, you must enter the physical table column mapped by the primary key properties. For example, if the ID card number is the primary key, you need to enter the ID card number when you add an ID card node in Analytics Workbench.</p>

Parameter	Description
Show in Graph	<p>If you select this parameter, Property Name of this property will be displayed in Graph in Analytics Workbench together with the object node. Otherwise, Property Name will not be displayed. For example, if you select the ID card number, this number will be displayed in Graph together with the ID card object. If you select the name property at the same time, the name and the ID card number will be displayed together with the ID card object.</p> <p>If you select this parameter for a specific property, a bubble icon appears next to the option. You can click the bubble icon to specify whether to show the Property Name in Graph. For example, if you select this parameter for the ID card number, and choose to display Property Name in Graph, the ID card property displayed in Graph will be ID card number: xxxxxx.</p>
Show in Properties	<p>If you select this parameter for a property, the property will be displayed on the Details tab and the Property tab in the right-side pane of the Graph page in Analytics Workbench. Otherwise, the property will not be displayed.</p>
Conditional Query:	<p>If you select this parameter for a property, you can query the object based on this property in Target Object when you perform an analysis on the Graph page.</p>
Show in Statistics	<p>If you select this parameter for a property, the property is displayed on the Statistics tab in the right-side pane of the Graph page in Analytics Workbench. If you do not select this parameter, the property is not displayed in Analytics Workbench.</p>
Available	<p>If you select this parameter for a property, the property takes effect and is displayed in Analytics Workbench. This parameter must be selected for primary key properties.</p> <p>If any of the following parameters has been selected for the property: Primary Key, Show in Graph, Show in Properties, Conditional Query and Show in Statistics, the Available parameter will be automatically selected for a property. The Available parameter will be automatically cleared if you clear all the preceding parameters.</p>
Display Type	<p>After you configure this parameter, the property is displayed on the Details tab and the Property tab in the right-side pane of the Graph page in Analytics Workbench based on the selected type.</p> <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note To display a property in the format of Dictionary, you need to configure a dictionary first.</p> </div>
Query Type	<p>The data type that is supported in the query condition of a property. If you select Dictionary for Display Type, you must select Dictionary Option for Query Type.</p>
Security Level	<p>The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.</p>

Parameter	Description
Search Item Configuration	Click the  icon in the Actions column corresponding to a property. Specify the following parameters: <ul style="list-style-type: none"> Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. Default Query Condition Settings: the default condition used for an object query. If other properties are used as conditions for a query, this condition is also used by default. Authorization Code: After the authorization code function is enabled, only the authorized users can access this property. Derived Property: After a property is set as a derived property, it is automatically generated based on other properties. Configure the method in which the field is generated as needed.
Default Query Condition Settings	
Authorization Code	
Derived Field	
Delete	If a property is no longer used, you can click the  icon to delete this property.
Sort order	The Move Up and Move Down arrows are used to adjust the order of properties that are displayed in Analytics Workbench.

- (Optional) If you need to add multiple properties, you can refer to the preceding steps to add more properties.
- (Optional) Specify parameters in **Advanced**.

[Parameters in Advanced](#) describes the parameters in **Advanced**.

Parameters in Advanced

Parameter	Description
Add Image To	The avatar of the object that is displayed in Graph . Select a property of the object, and then specify the URL of the image and the suffix of the image. Add Image To allows you to specify a combination of the prefix, the property, and the suffix. The prefix is the URL of the image, and the suffix is the image format.
Logical Relation of Authorized Properties	The logical relationship between the authorization codes of properties in each record: <ul style="list-style-type: none"> AND: The current record is visible to the users who meet all authorization code conditions of the properties in this record. OR: The current record is visible to the users who meet any one authorization code condition of the properties in this record.

- Click **Save**.

8.5.2.3. Enable and disable an object

After you have created a complete object (configured the object properties), the object is enabled automatically. You can disable an object if it is no longer used for a certain period of time and enable it again when necessary.

Prerequisites

To disable objects, you must first delete the dependency information of these objects, including the mappings between the objects and data tables and the objects referenced by links.

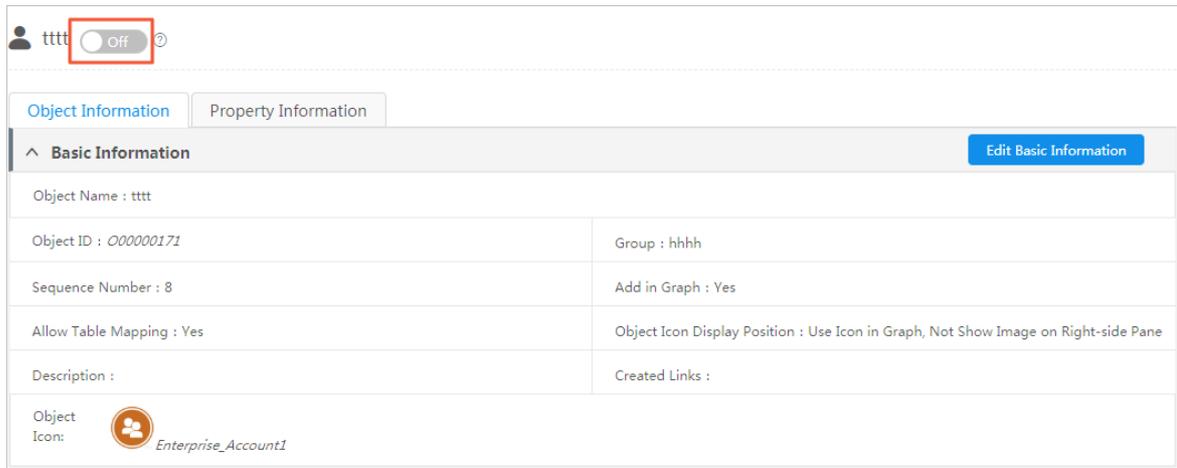
Context

You cannot use an object in Analytics Workbench after the object is disabled.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane, click the object group to which the object belongs, and then click the object to be enabled or disabled.

The detailed information of the object is displayed on the right side of the page.



4. Enable or disable an object as follows.

Operation	Procedure
Disable an object	If the current object is enabled, you can click the  icon to disable the object. After the object is disabled, the Status changes from a highlighted  icon to a gray  icon.
Enable an object	If the current object is disabled, you can click the  icon to enable the object. After the object is enabled, the Status changes from a gray  icon to a highlighted  icon.

8.5.2.4. Modify an object

In Graph Analytics, you can modify the basic information of an object at any time, including the object name, object group, and object icon.

Prerequisites

You have created an object. For more information about how to create an object, see [Create an object](#).

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane, click the object group to which the object belongs, and then click the object to be modified.
4. In right-side area, click **Edit Basic Information**.

The screenshot shows the configuration page for an object named 'SV_Account'. At the top, there is a user profile 'SV_Account' with a toggle switch set to 'On'. Below this are two tabs: 'Object Information' (selected) and 'Property Information'. Under the 'Object Information' tab, there is a section titled 'Basic Information' with an expandable arrow. To the right of this section is a blue button labeled 'Edit Basic Information', which is highlighted with a red border. Below the 'Basic Information' section, there is a table of object properties:

Object Name : SV_Account	
Object ID : 00000072	Group : finance
Sequence Number : 1	Add in Graph : Yes
Allow Table Mapping : Yes	Object Icon Display Position : Use Image in Graph, Not Show Image on Right-side Pane
Description : SV_Account	Created Links : SV_Transfer_Link
Object Icon:  BankCard	

5. Modify the object information as needed.
All parameters can be modified except the **Object ID**.
6. After you have configured the preceding parameters, click **Save**.

8.5.2.5. Delete an object

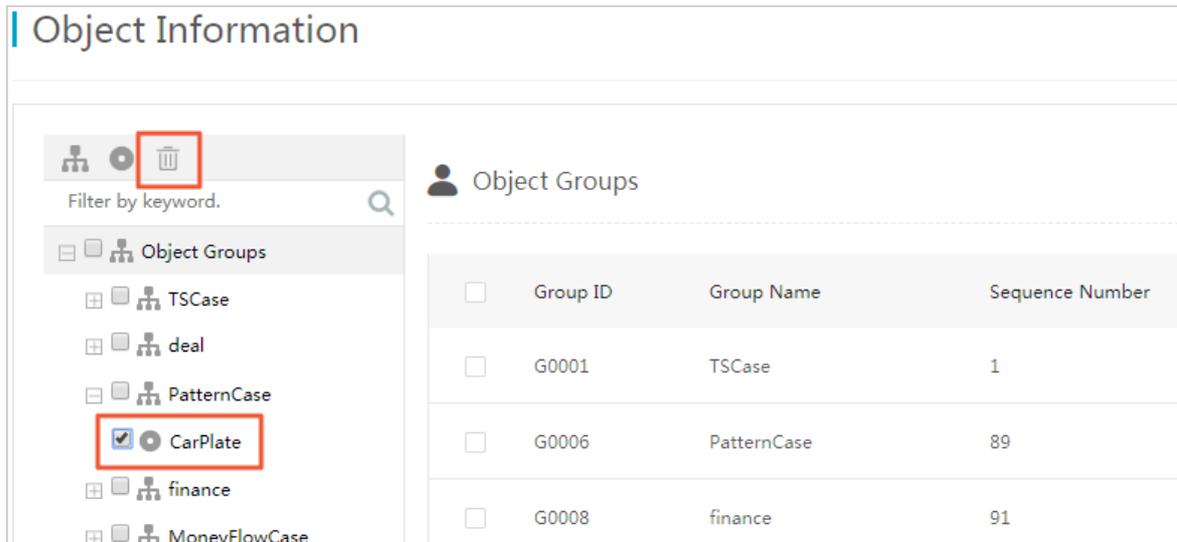
You can delete the objects that are no longer used.

Prerequisites

You have deleted the mappings between the object and the data table. You have deleted links and events that are referenced by the object.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane, click the object group to which the object belongs, and select one or more objects to be deleted. You can open multiple object groups and select objects from different groups.



4. Select the objects to be deleted, and click the  icon.
5. In the Delete Object Information dialog box that appears, click **OK**.

8.6. Link information

8.6.1. Link groups and links

8.6.1.1. Create a link group

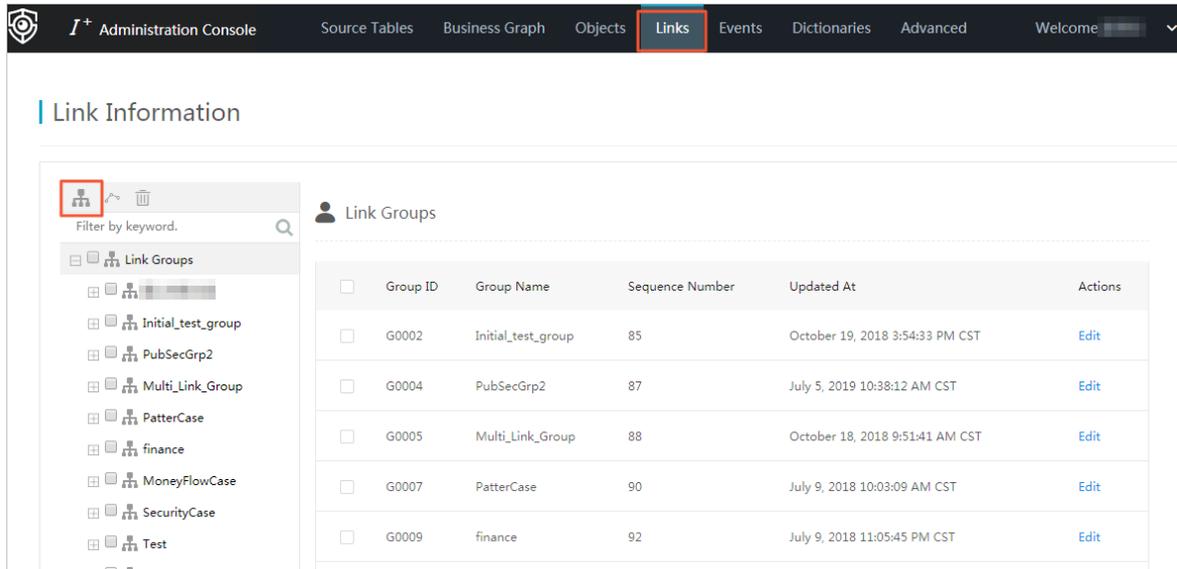
You can use link groups to classify links, so that you can search for and manage links with ease. One link must be and can be included in only one link group. You need to create a proper link group before you create a link.

Prerequisites

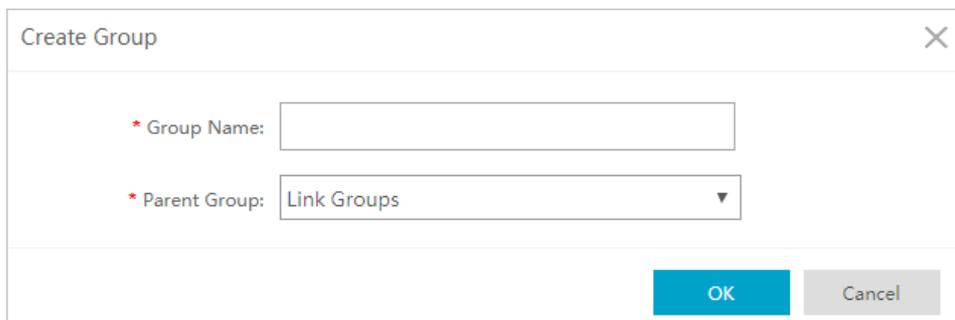
You have an account and password for Graph Analytics and have permissions to perform operations in Administration Console of Graph Analytics.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Links**.



3. On the **Link Information** page, click the **Create Group** icon .
4. In the **Create Group** dialog box that appears, specify **Group Name** and **Parent Group**.



5. Click **OK**.

8.6.1.2. View links and link groups

Graph Analytics allows you to view all link groups in the current environment so that you can understand the existing link groups and the links under each group at any time.

Prerequisites

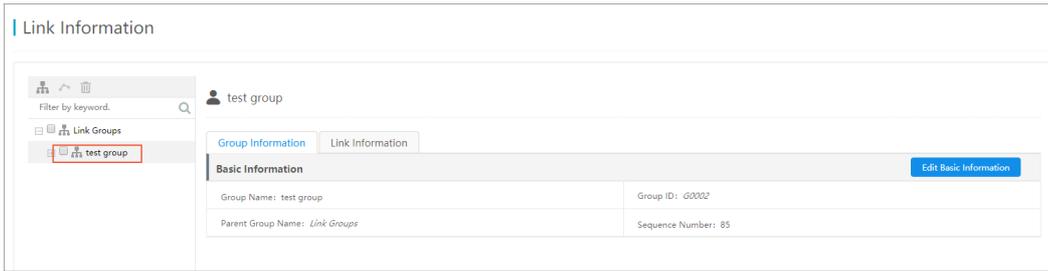
You have an account and password for Graph Analytics and have permissions to perform operations in Administration Console of Graph Analytics.

Procedure

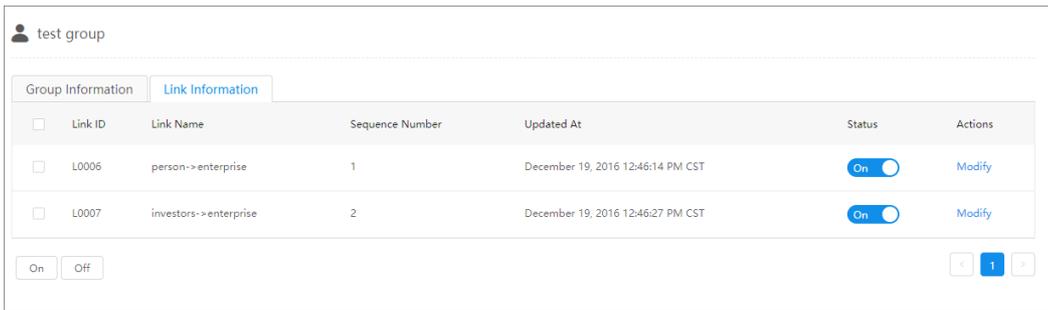
1. **Log on to Administration Console of Graph Analytics.**
2. In the top navigation bar, click **Links**.
3. In the left-side navigation pane of the **Link Information** page, click a link group. The **Group Information** and **Link Information** tabs of the link group are displayed on the right side of the page.

The left-side navigation pane displays all link groups in the current environment. You can view the groups one by one.

Group Information



Link Information



8.6.1.3. Modify a link or link group

In Graph Analytics, you can modify the name and order number of a link or link group. You can adjust the basic information of a link or link group at any time as needed. In Graph Analytics, you can enable or disable a link.

Prerequisites

- You have created a link group. For more information about how to create a link group, see [Create a link group](#).
- You have created a link that belongs to this link group. For more information about how to create a link, see [Create a first-degree link](#), [Create a second-degree link](#), or [Create a multi-degree link](#).
- To disable links, you must first delete the dependency information of these links, including the mappings between the links and data tables.

Context

This topic describes the following operations:

- Modify the basic information about a link group
- Modify the basic information about a link
- Enable or disable a link

After you create a complete link by configuring the link properties, the link is automatically enabled. You can disable a link if it is no longer used for a period of time and enable it again when necessary. You cannot use a link in Analytics Workbench after the link is disabled.

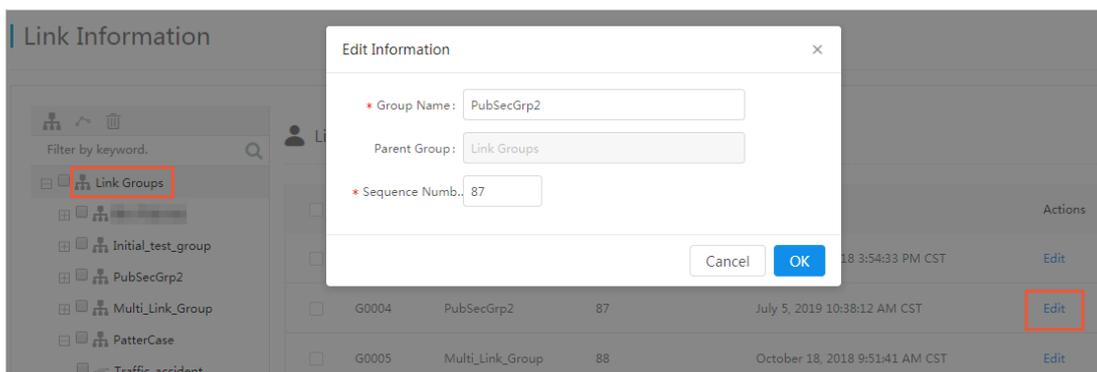
Modify the basic information about a link group

You can modify the basic information of an object group by using one of the following methods:

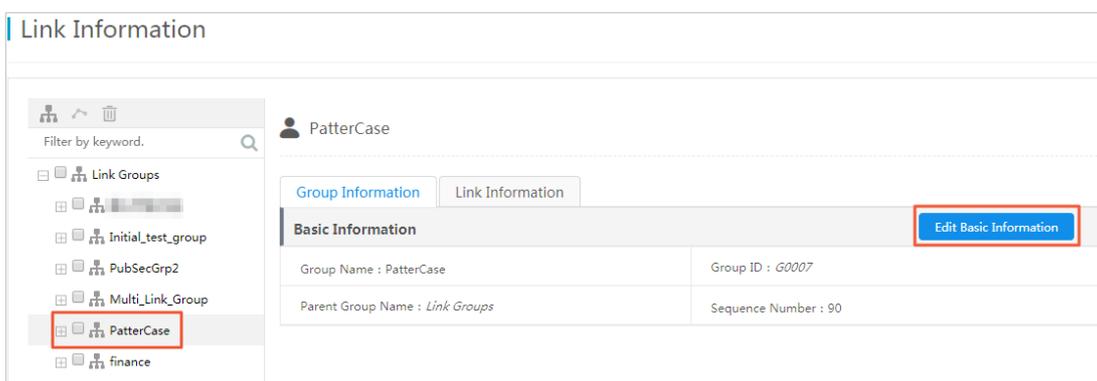
Configuration method	Procedure
----------------------	-----------

Configuration method	Procedure
Method 1	<ol style="list-style-type: none"> 1. Log on to Administration Console of Graph Analytics. 2. In the top navigation bar, click Links. 3. In the Link Groups area that appears, select a link and click Edit in the Actions column. 4. In the Edit Information dialog box that appears, specify Group Name and Sequence Number, as shown in Edit the basic information. 5. Click OK.
Method 2	<ol style="list-style-type: none"> 1. Log on to Administration Console of Graph Analytics. 2. In the top navigation bar, click Links. 3. In the left-side navigation pane, click the link group you want to modify. 4. On the right-side Group Information tab, click Edit Basic Information. Specify Group Name and Sequence Number, as shown in Edit the basic information. 5. Click Save.

Edit information

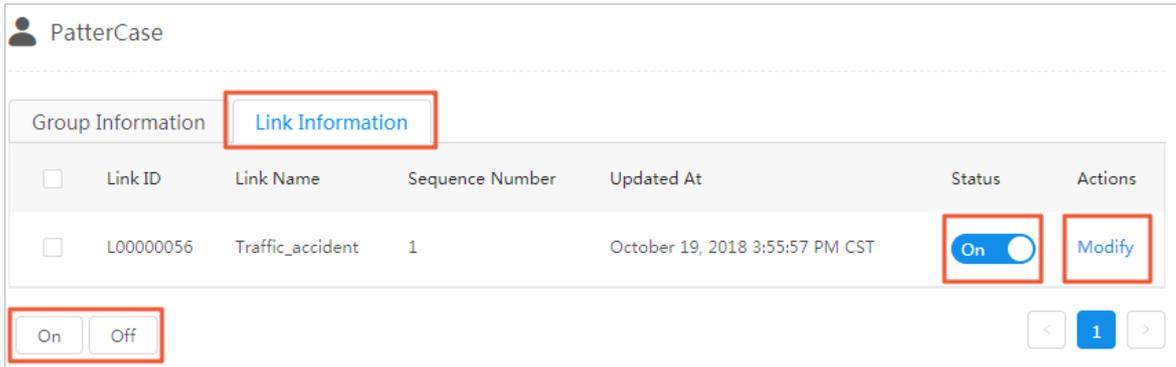


Edit the basic information



Modify the basic information about a link

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, click **Links**.
3. In the left-side navigation pane, click the link group to which the link belongs, and then click the **Link Information** tab.



4. In the **Edit Information** dialog box that appears, specify **Link Name** and **Sequence Number**.
5. Click **OK**.

Enable or disable a link

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.
3. In the left-side navigation pane, click the link group to which the link belongs, and then click the **Link Information** tab on the right side of the page.
4. You can use the following methods to enable or disable a link on the **Link Information** tab.

Configuration method	Procedure
Disable a link	<p>Disable a single link: Click the On icon next to the link you want to disable.</p> <p>Disable multiple links: Select the links you want to disable and then click Off in the lower part of the page.</p> <p>After the link is disabled, the Status changes from a highlighted On icon to a dimmed Off icon.</p>
Enable a link	<p>Enable a single link: Click the Off icon next to the link you want to enable.</p> <p>Enable multiple links: Select the links you want to enable and then click On.</p> <p>After the link is enabled, the Status changes from a dimmed Off icon to a highlighted On icon.</p>

8.6.1.4. Delete a link or link group

You can delete the links and link groups when they are no longer used.

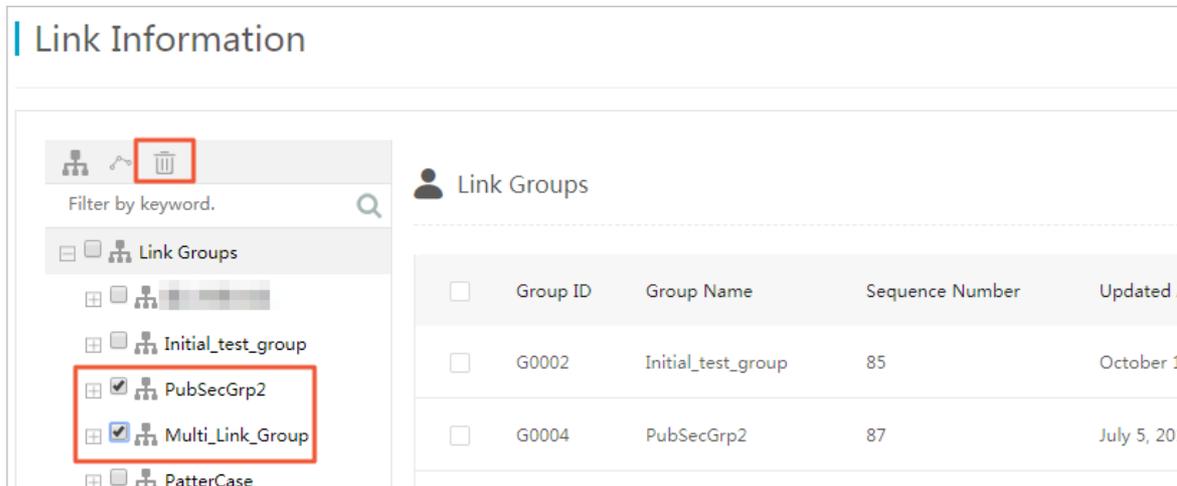
Prerequisites

- Before you delete a link, you have deleted the dependency information of the link, for example, the mapping between the link and the physical table.
- Before you delete a link group, you have deleted all the links in the group. You can delete only empty link groups.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.

3. (Optional) If there are links in the object group, you must delete these links first.
 - i. In the left-side navigation pane, select all links in the link group you want to delete, and click the  icon in the upper-left corner.
 - ii. In the **Delete Link Information** dialog box that appears, click **OK**.
4. In the left-side navigation pane, select the link groups you want to delete, and then click the  icon in the upper-left corner.



5. In the **Delete Link Information** dialog box that appears, click **OK**.

8.6.2. First-degree links

8.6.2.1. Create a first-degree link

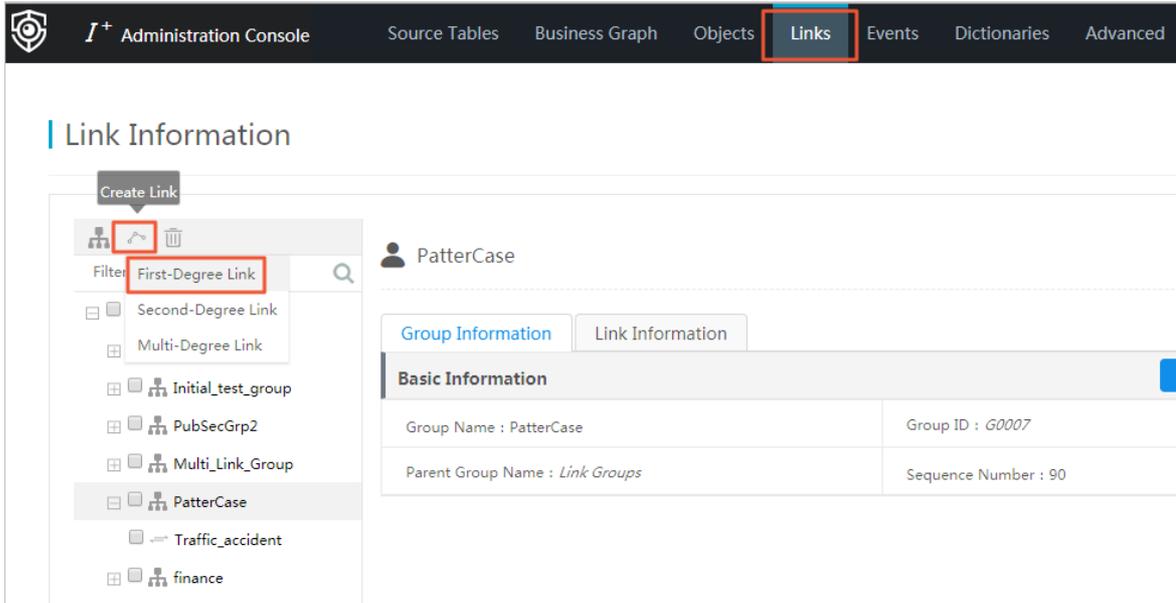
A first-degree link reflects the direct relationship between two objects. It is the basis of second-degree links and multiple-degree links. You must create a first-degree link between objects before you perform a link analysis in Graph Analytics. A complete link contains the basic information, property information, and business parameters. This topic describes how to configure the basic information of a link.

Prerequisites

- You have created and enabled a source object and a target object for the first-degree link. For more information about how to create an object, see [Create an object](#).
- You have created a link group to which the first-degree link belongs. For more information about how to create a link group, see [Create a link group](#).

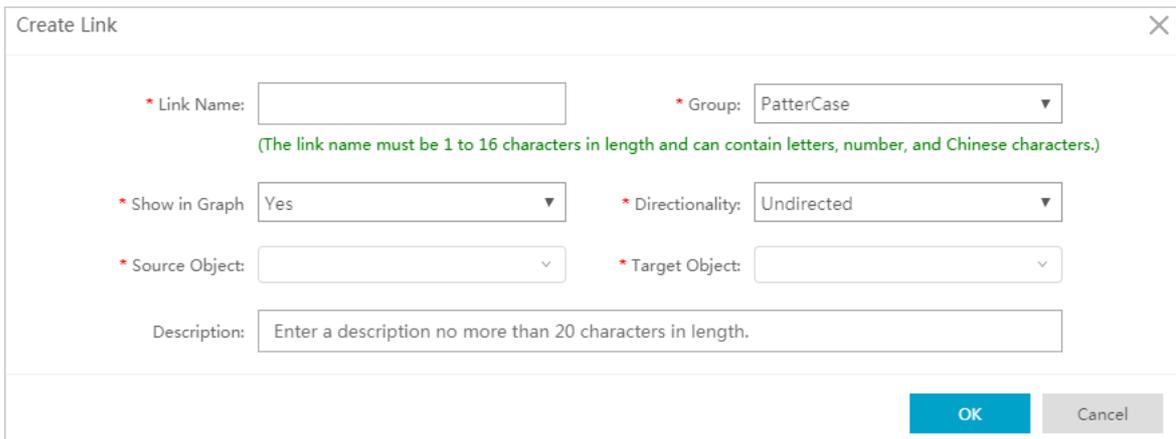
Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.
3. The **Link Information** page appears. Click the **Create Link** icon, and click **First-Degree Link**.



4. In the **Create Link** dialog box that appears, specify the parameters.

Parameters used to create a first-degree link describes the parameters.



Parameters used to create a first-degree link

Parameter	Description
Link Name	The user-defined link name. It must be unique.
Group	By default, the selected link group in the left-side navigation pane is used. You can also select another link group as needed.
Show in Graph	Specifies whether a link is displayed on the Graph page. If the value is set to No , this link is not displayed on the Graph page.
Directionality	Specifies whether the link is directional. For example, if the source object A calls the target object B, a link is established. If this link is set to be directional, it will be displayed on the Graph page. The link direction is from A to B (A > B).
Source Object	The source object of the first-degree link. You can select it from the drop-down list.

Parameter	Description
Target Object	The target object of the first-degree link. You can select it from the drop-down list.
Description	Enter the description of the first-degree link. This makes it easy for you to understand this link.

5. Click **OK**.

What's next

1. After you have created a first-degree link, you must specify the properties and configure the business parameters based on your requirements. For more information, see [Configure link properties and business parameters](#).
2. To use the new link, you must log on to Analytics Workbench again.

8.6.2.2. Configure link properties and business parameters

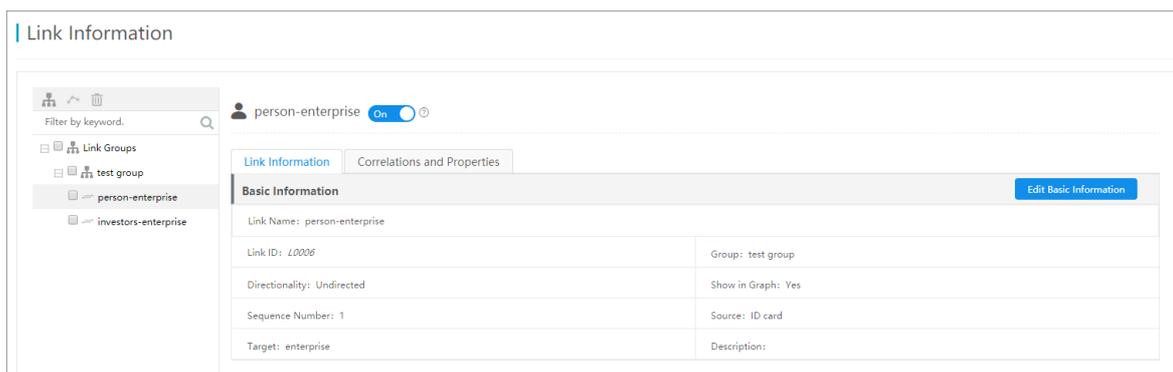
After a first-degree link is added, you need to configure the properties and business parameters of the link based on your business requirements, so that you can view and apply this link in Analytics Workbench. This topic describes how to configure the properties and business parameters of a first-degree link.

Prerequisites

A first-degree link is created. For more information, see [Create a first-degree link](#).

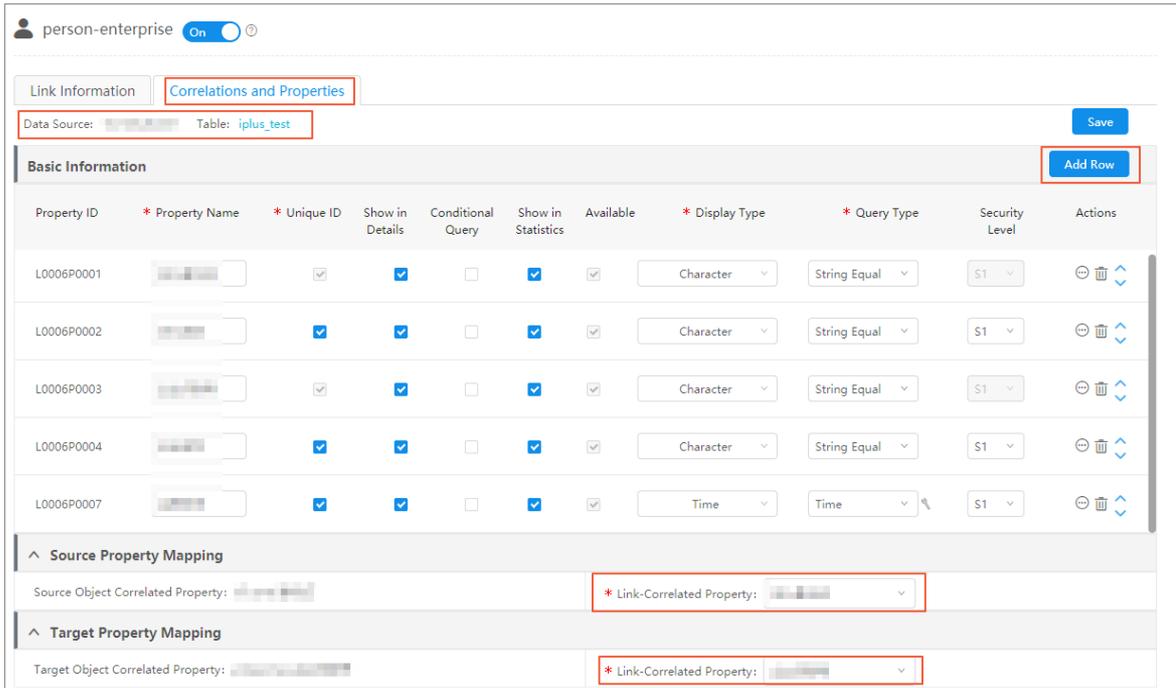
Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.
3. In the left-side navigation pane, click the link group that contains the first-degree link you want to configure, and then click the target link.



4. On the right side of the page, click the **Correlations and Properties** tab.

On the **Correlations and Properties** tab, if the link is mapped to a data table in the data source, **Data Source** and **Table** are displayed in the property information above the Basic Information section. You can click the table name to redirect to the table page.



5. Configure the parameters in the **Basic Information** section.

If you need to add a property for a link in the **Basic Information** section, click **Add Row**.

Note

Link-Correlated Property in the **Source Property Mapping** section and Link-Correlated Property in the **Target Property Mapping** section must be different and both of them must be configured. Therefore, a link must have two or more properties.

Parameters in the Basic Information section describes the parameters in the **Basic Information** section.

Parameters in the **Basic Information** section

Parameter	Description
Property ID	The ID of a property. It is automatically generated.
Property Name	The name of a property. If you select Available for a property, the name of the property is displayed in Analytics Workbench.
Unique ID	Allows you to define the logical primary key of a data table.
Show in Details	If you select this parameter for a property, the property is displayed on the Details tab in Analytics Workbench.
Conditional Query	If you select this parameter for a property, you can query a link based on the link type specified in Link Type when you perform analysis on the Graph page in Analytics Workbench.
Statistics	If you select this parameter for a property, the property is displayed on the Statistics tab in the right-side pane of the Graph page in Analytics Workbench. If you do not select this parameter, the property is not displayed in Analytics Workbench.

Parameter	Description
Available	<p>If you select this parameter for a property, the property takes effect and is displayed in Analytics Workbench.</p> <p>If one of the following parameters is selected for the property, the Available option is automatically selected for a property: Unique ID, Show in Details, Conditional Query, and Show in Statistics. If all of the preceding parameters are cleared for a property, the Available parameter is automatically cleared for the property.</p>
Display Type	<p>After you set the display type, the property is displayed on the Property tab in the right-side pane of the Graph page in Analytics Workbench based on the selected type.</p> <div style="background-color: #e1f5fe; padding: 5px; border: 1px solid #ccc;"> <p> Note If you want to display a property in the Dictionary format, configure a dictionary first.</p> </div>
Query Type	<p>The data type that is supported in the query condition of a property. If you select Dictionary for Display Type, you must select Dictionary Option for Query Type.</p>
Security Level	<p>The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.</p>
Search Item Configuration	<p>Find the property and click the More icon  in the Actions column. Configure the following parameters:</p> <ul style="list-style-type: none"> ◦ Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. ◦ Default Query Condition Settings: the default condition used for a link query. If other properties are used as conditions for a query, this condition is also used by default. ◦ Authorization Code: After the authorization code function is enabled, only authorized users can access this property. ◦ Derived Property: After a property is configured as a derived property, it can be generated automatically based on other properties. Configure the method in which the column is generated based on your needs.
Default Query Condition Settings	
Authorization Code	
Derived Field	
Delete	<p>If a property is no longer used, you can click the Delete icon  to delete this property.</p>
Sort order	<p>The Move Up and Move Down arrows are used to adjust the order of properties that are displayed in Analytics Workbench.</p>

6. Set Link-Correlated Property in the Source Property Mapping and Target Property Mapping sections.

These configurations are related to Source Object and Target Object of the link.

Use **Source Property Mapping** as an example. **Source Object Correlated Property** and **Link-Correlated Property** must be mapped to the same column in the same table. The **Source Object Correlated Property** parameter is the primary key property of the source object, which is automatically loaded based on the Source Object parameter. For the **Link-Correlated Property** parameter, you must select the link property that is mapped to the same column in the same table as the Source Object primary key.

Set **Target Property Mapping** in the same way you set **Source Property Mapping**.

- (Optional) Configure parameters in the **Advanced**, **Accumulative Statistics Settings**, and **Link Weight Settings** sections based on your requirements.

For more information about the key parameters, see [Key parameters](#).

The screenshot shows a configuration interface with three main sections:

- Advanced:** Contains dropdown menus for Chronological Time Property, Time Property for Behavior Analysis, Linked Times, Details Sorting Property, Logical Relation of Authorized Properties, and Link Name Definition (set to Link Name and person-enterprise).
- Accumulative Statistics Settings:** Includes a blue 'Set Linked Times' button and a grey 'Add Statistical Condition' button.
- Link Weight Settings:** Includes a checkbox for 'Configure Link Weight', a 'Weight Property' dropdown, radio buttons for 'Sequence' (Ascending selected, Descending), 'Weight Thresholds' input fields, and a note 'Weight Segments from Low to High (-∞, +∞)'. The 'Configure Link Weight' checkbox is currently unchecked.

Key parameters

Section	Parameter	Description
Advanced	Chronological Time Property	The link properties involved in chronological analysis. From the drop-down list, select one or more link properties with the query type of time.
	Time Property for Behavior Analysis	The link properties involved in behavior analysis. From the drop-down list, select one or more link properties with the query type of time.
	Linked Times	The number of the same values that are counted for a specific property in a link. The total number is displayed as the number of link occurrences. The Linked Times parameter is used as the default setting to filter link types. Assume that two lines of calls from user A to user C are displayed in the call log. The number of calls from user A to user C is displayed as two in the analysis results.
	Details Sorting Property	The default property by which the returned behavior details are sorted.

Section	Parameter	Description
Accumulative Statistics Settings	None	Used to perform logical statistics for link properties of which the query type is numeric range. The logical statistics operations include top, equal sign (=), and greater-than-or-equal-to sign (\geq). This configuration is suitable for business scenarios where statistics filter is required for link query results. The Linked Times parameter is used to filter records in the link query results. You can add statistical conditions to filter the link properties with the query type of numeric range.
Link Weight Settings	None	You can specify a link property with the query type of numeric range and calculate the link weight based on the numeric range specified for the link property.

- After you modify the parameter configurations, click **Save**. A message appears, indicating that the modifications are saved.

8.6.3. Create a second-degree link

A second-degree link reflects the relationship between two objects established by using an intermediary object. Unlike a first-degree link, a second-degree link can be used to explore a more complex relationship network among objects.

Prerequisites

- Related first-degree links have been created for the second-degree link. For more information about how to create a first-degree link, see [Create a first-degree link](#).
- A link group has been created for the second-degree link. For more information about how to create a link group, see [Create a link group](#).

Context

A second-degree link is created based on first-degree links and can be split into two first-degree links. For example, object A has a first-degree link with object C, and object B also has a first-degree link with object C. In this case, you can establish a second-degree link between object A and object B. Typically, people taking the same train or plane or staying in the same hotel have a second-degree link and can be analyzed by using the second-degree link model.

Procedure

- [Log on to Administration Console of Graph Analytics](#).
- In the top navigation bar, click **Links**.
- On the **Link Information** page that appears, click the **Create Link** icon and select **Second-Degree Link**.
- Configure the parameters on the **Link Definition** page.
[Parameters](#) describes the parameters.

Call_second_link_01 Off ?

1 Link Definition
2 Link Element Definition
3 Link Computing Configuration

* Link Name:

(The link name must be 1 to 16 characters in length and can contain letters, number, and Chinese characters.)

Link ID:

* Group:

* Show in Graph:

* Source Object:

* Target Object:

* First-Degree Link:

Description:

Parameters

Parameter	Description
Link Name	The user-defined link name. It must be unique.
Group	By default, the selected link group in the left-side navigation pane is used. You can also select another link group as needed.
Show in Graph	Indicates whether a link can be displayed on the Graph page. If this parameter is set to No , this link is not displayed on the Graph page.
Source Object	The source object of the second-degree link.
Target Object	The target object of the second-degree link.
First-Degree Link	The first-degree links between the source object and the target object. These links can be used to create a second-degree link.
Description	The description of the second-degree link that helps you understand the link.

5. After you have configured the preceding parameters, click **Next** to configure the parameters in **Link Element Definition**.

[Parameters](#) describes the parameters.

Call_second_link_01 Off ⓘ

1 Link Definition — 2 **Link Element Definition** — 3 Link Computing Configuration

Define Basic Query Conditions Select Query Properties

Basic Query Condition ID	Query Property	Query Method	Default Query Value	Actions
undefinedC0001	caller_num	Equal Value	<input type="text"/>	Delete ⌵
undefinedC0002	callee_num	Equal Value	<input type="text"/>	Delete ⌵

Define Base Link Select Link Properties Select Object Properties

Base Link ID	Base Link Name	Referenced First-Degree Link Property	Correlation Rule	Default Value	Actions
undefinedC0003	<input type="text" value="caller_numSame"/>	caller_num	Equal Value	<input type="text"/>	Delete

Previous Next Submit

Parameters

Section	Parameter	Description
Define Basic Query Conditions	Select Query Properties	Allows you to define the basic query conditions that can be used to query a link. You can view the conditions in the Basic Properties section in the Link Type settings when you configure link extension.
Define Basic Link	Select Link Properties and Select Object Properties	Allows you to define the table columns that are needed to perform a query. You can define the alias of a column in Base Link Name . The base link can be referenced in Link Configuration in the next step. Configure the Correlation Rule. The Default Value will be used when you perform a query in Analytics Workbench.

- After you have configured these parameters, click **Next** and then configure the parameters in **Link Computing Configuration**.

[Parameters](#) describes the parameters.

Call_second_link_01 Off ⓘ

1 Link Definition — 2 Link Element Definition — 3 **Link Computing Configuration**

Link Configuration Add Link Configuration

Link ID	Link Name	Referenced Base Links	Actions
No data			

Link Sets Add Link Set

Set ID	Set Name	Content	Link Selected by Default	Default Times:	Actions
No data					

Previous Next Submit

Parameters

Section	Parameter	Description
Link Configuration	Add Link Configuration	<p>Allows you to combine multiple first-degree links to form a link that meets multiple conditions.</p> <ol style="list-style-type: none"> Click Add Link Configuration, specify Link Name, and then select links. Click OK to add a link that meets multiple conditions. You can repeat the preceding steps to create more links of the same type as needed.
Link Sets	Add Link Set	<p>A link set is a group of links configured in Link Configuration. Such link sets define the advanced query conditions for link queries in link analysis.</p> <ol style="list-style-type: none"> Click Add Link Set and configure the parameters. <p>Key parameters to add a link set:</p> <ul style="list-style-type: none"> ■ Default Times: the minimum number of occurrences of a link set that can be counted as a query match. The default value is 2. ■ Base Link: the links displayed in the Select Base Links area are all the links that have been configured in Link Configuration. Each link can be contained in only one link set. ■ Link Selected by Default: the default query condition that is displayed on Analytics Workbench. If a link set that contains multiple links is used for a link query, each analysis is performed based on one of the links. By default, Link Selected by Default is used. <ol style="list-style-type: none"> Click OK. You can repeat the preceding steps to create more link sets as needed.

Add a link configuration

Add a link set

The screenshot shows a dialog box titled "Add Link Set". At the top right is a close button (X). Below the title bar, there is a "Default Times:" label followed by a text input field containing the number "2". Underneath is a section labeled "Select Base Links:" containing a table. The table has three columns: "Base Link", "Link Selected by Default", and "Sort". One row is visible with "Caller_num_s" in the "Base Link" column, a checked checkbox in the "Link Selected by Default" column, and a sort icon (up and down arrows) in the "Sort" column. Below the table, the word "ame" is partially visible. At the bottom right of the dialog are "Cancel" and "OK" buttons.

7. After you have configured the preceding parameters, click **Submit** to create a second-degree link.

8.6.4. Create a multi-degree link

A multi-degree link reflects the relationship between two objects established by using multiple intermediary objects. Compared to a first-degree link or a second-degree link, a multi-degree link can further explore the complex relationship between objects.

Prerequisites

- You have created first-degree links for the multi-degree link. For more information about how to create a first-degree link, see [Create a first-degree link](#).
- You have created second-degree links for the multi-degree link. For more information about how to create a second-degree link, see [Create a second-degree link](#).
- You have created a link group for the multi-degree link. For more information about how to create a link group, see [Create a link group](#).

Context

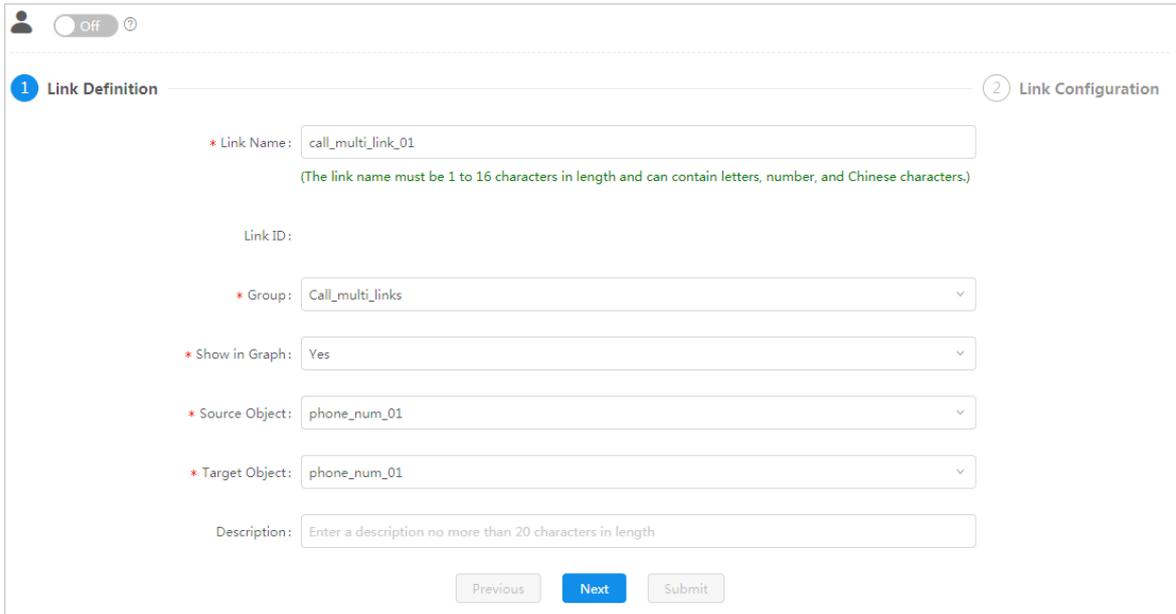
A multi-degree link uses multiple first-degree links and second-degree links to query the relationship between two objects.

For example, if mobile phone A and mobile phone B do not have call or text message records and only have email correspondence records, there are only indirect links between the two mobile phones. In this example, there is no direct relationship between mobile phone A and mobile phone B, so they cannot be directly linked. When mobile phone A sends an email to mobile phone B, an indirect link is established. This indirect link involves three first-level links: the link between mobile phone A and mailbox A, the link between mailbox A and mailbox B, and the link between mobile phone B and mailbox B. You can use mobile phone A to query mobile phone B by using the three links. Indirect links can also be established between two mobile phones that are communicated by using apps such as Facebook and Twitter.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Links**.
3. On the **Link Information** page that appears, click the **Create Link** icon and select **Multi-Degree Link**.
4. Configure the parameters on the **Link Definition** page.

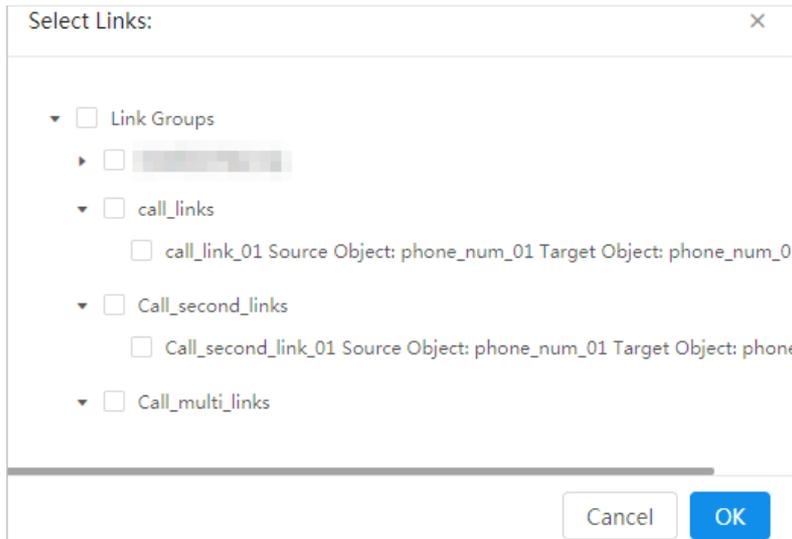
[Link definition parameters](#) describes the parameters.



Link definition parameters

Parameter	Description
Link Name	The user-defined link name. It must be unique.
Group	By default, the selected link group in the left-side navigation pane is used. You can also select another link group as needed.
Show in Graph	Specifies whether a link is displayed on the Graph page. If the value is set to No , this link definition is not displayed on the Graph page.
Source Object	The source object of the multi-degree link. You can select it from the drop-down list.
Target Object	The target object of the multi-degree link. You can select it from the drop-down list.
Description	The description of the multi-degree link that helps you understand the link.

5. Click **Next**.
6. On the **Link Configuration** page, click **Select Links** in the upper-right corner, and then select the related first-degree links and second-degree links.

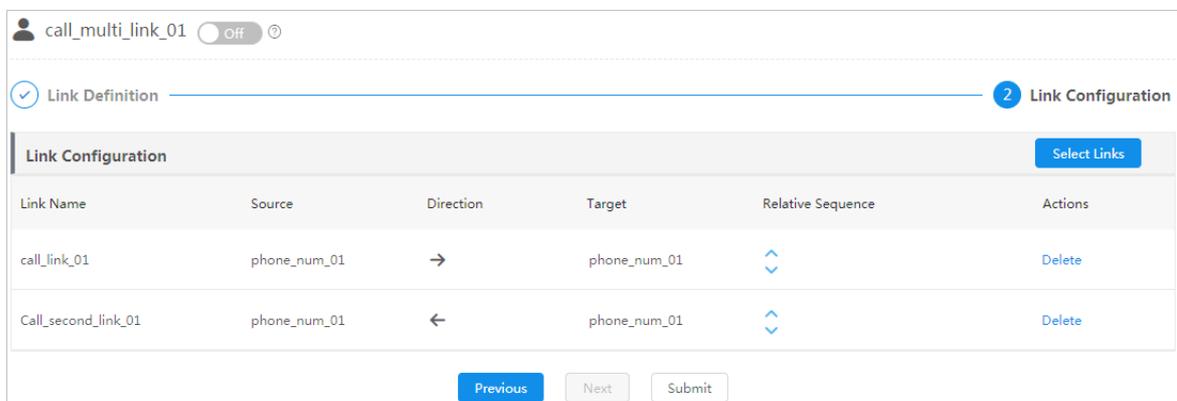


7. After you have configured the preceding parameters, click **OK** to add first-degree links and second-degree links.

In most cases, you may need to add two first-degree or second-degree links with opposite query directions. For example, if you want to send an email by using your mobile phone, the query direction is from the mobile phone to the email box. If you want to receive an email on your mobile phone, the query direction is from the email box to your mobile phone. Therefore, two first-degree links can be added with opposite query directions.

8. On the **Link Configuration** page, click the right or left arrow in the **Query Direction** column to adjust the query direction.

The source object and the target object must be correlated with each other by using intermediary objects..



9. After you have configured the preceding parameters, click **Submit** to add a multi-degree link.

8.7. Event information

8.7.1. Event groups

8.7.1.1. Create an event group

You can use event groups to classify events, so that you can easily find and manage events. Any event must be and can only be grouped into one event group. You need to create a proper event group before you create an event.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. At the top of the left-side navigation pane, click the **Create Group** icon . The **Create Group** dialog box appears.
4. Enter a **Group Name** as needed, and then select **Event Groups** from the *Parent Group* drop-down list.
5. Click OK.

8.7.1.2. View an event group

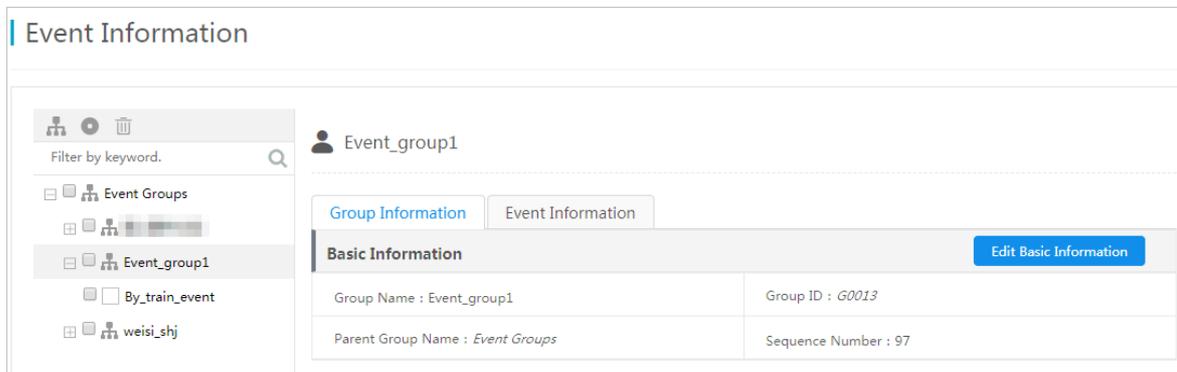
You can view the basic information of an event group and the events in the group in Administration Console.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. From the left-side navigation pane, select an event group such as **By Train**. The detailed information about this event group is displayed on the right side of the page.



You can view event group information on the **Group Information** and **Event Information** tabs. The **Group Information** tab displays the basic information about the event group. The **Event Information** tab displays all events in the group.

8.7.1.3. Modify an event group

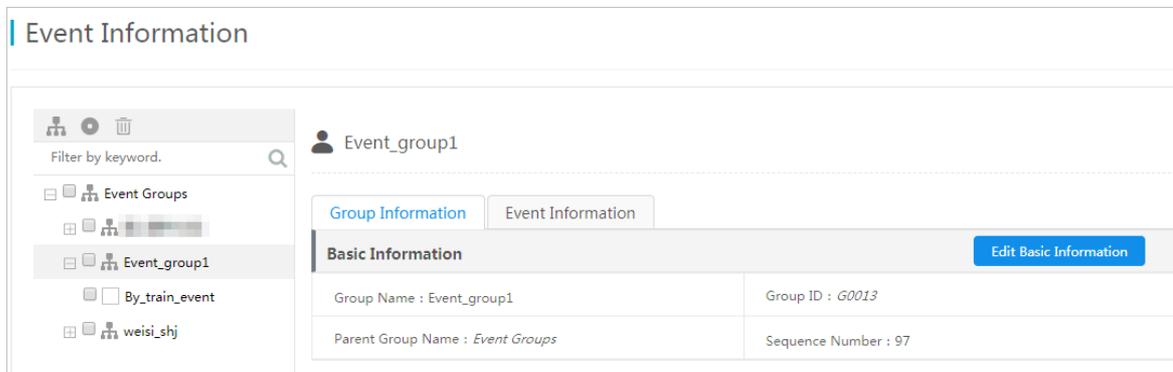
You can modify the basic information of an event group and the events in the group in Administration Console.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. From the left-side navigation pane, select an event group such as **By Train**. The detailed information about this event group is displayed on the right side of the page.



4. Click **Edit Basic Information** on the **Group Information** page. You can then modify the **Group Name** and **Sequence Number** of the group.
5. Click **Save**.

8.7.1.4. Delete an event group

If an event group is no longer used, you can first delete all events in the group and then delete the event group.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have deleted all events in the event group. For more information about how to delete an event, see [Delete an event](#).

Procedure

1. Log on to Administration Console of Graph Analytics.
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. Select the event groups that you want to delete from the left-side navigation pane, and then click the **Delete** icon  on the top of the pane.
4. In the message box that appears, click **OK**.

8.7.2. Events

8.7.2.1. Create an event

Events are used to analyze the behaviors of objects. Event information is used to define the event models in Graph Analytics. A complete event contains basic event information, properties, and relevant parameters. This topic describes how to configure the basic information of an event.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on

Analytics Workbench and Administration Console.

- You have created an event group for the event. For more information about how to create an event group, see [Create an event group](#).

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. On the top of the left-side navigation pane, click the **Create Event** icon . Set the parameters in the **Create Event** dialog box. For more information about the parameters and descriptions, see [Parameters of created events](#). Parameters of created events

Parameter	Configuration method
Event Name	The name of the event. Set this parameter as needed. Each event name must be from 1 to 16 characters in length, and can contain letters, digits, and Chinese characters.
Event Description	The description of the event. Set this parameter as needed to help users understand the event.
Group	The event group that the event belongs to. Set this parameter as needed.
Event Icon	The event icon. You can select an icon in Icon Library , or enter an accessible URL to reference an external icon.

4. After you set the parameters, click **OK**. A message is displayed, indicating that the event has been created.

What's next

1. After you have created an event, you must configure the properties and business parameters based on your business requirements. For more information, see [Configure event property parameters](#).
2. After you have configured the properties and business parameters of an event, you must log on to Analytics Workbench again to use the new event.

8.7.2.2. Configure event properties and business parameters

After you have created an event, you must configure the event properties. Event properties are critical to an event. You can configure event properties, and associate the properties with objects on the **Property Information** tab.

Prerequisites

You have obtained an account and a password for Graph Analytics and you are granted the required **event permissions**.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. Click an event in the left-side navigation pane, and click the **Property Information** tab on the right side of the page. The **Property Information** tab appears.

4. Set the required event parameters.

Click **Add Row** to add a property for a link in **Basic information**.

The required parameters are displayed in the **Basic Information** and **Set Mappings Between Correlated Objects and Properties** areas. [Required event property parameters](#) describes the event parameters.

 **Note** To save the property information, you must set all the required parameters.

Required event property parameters

Category	Parameter	Description
	Property ID	The ID of a property. It is automatically generated.
	Property Name	The property name that is displayed on Analytics Workbench. We recommend that you enter a name that describes the business type.
	Primary key	Each primary key uniquely identifies an event. A property cannot be deleted after it has been configured as a primary key.
	Show in Graph	<p> Note For each event, you must select one or more properties you want to display in the graph.</p> <p>If you select this parameter, the Property Name of this property will be displayed in Graph in Analytics Workbench together with the object node. Otherwise, the property is not displayed. For example, if this parameter is selected for the ID card property of an event, the ID card number will be displayed in Graph with the event. If the name property is also selected, the name and the ID card number will be displayed with the event.</p> <p>If you select this parameter for a specific property, a bubble icon appears next to the option. You can click the bubble icon to specify whether to show the Property Name in Graph. For example, if you select this parameter for an ID card number, and choose to display the Property Name in Graph, the ID card property displayed in Graph will be ID card number: xxxxxx.</p>
	Show in Properties	If you select this parameter for a property, the property will be displayed in the right-side pane on the Graph page of Analytics Workbench. To view the property, choose Properties > Event Properties .
	Element Identifier	The element identifier of a property. Set this parameter based on the actual property.
	Conditional Query	If you select this parameter for a property, the event can be queried based on this property in Link Type on the Graph page of Analytics Workbench when you perform a relationship analysis.

Category	Parameter	Description
Basic information	Statistics	If you select this parameter for a property, the event will be displayed in the right-side pane on the Graph page of Analytics Workbench. To view the property, choose Statistics > Event Distribution .
	Available	If you select this parameter for a property, the property takes effect and can be displayed on the Graph page of Analytics Workbench. This parameter is selected by default and cannot be changed.
	Display Type	The format in which a property is displayed in the right-side pane of the Graph page on Analytics Workbench. Set this parameter as needed.
	Query Type	The data type that is supported in the query condition of a property.
	Security Level	The security level for a property. A user with a security level lower than the value of this parameter cannot view the property.
	Search Item Configuration	Click the More icon  in the Actions column that corresponds to a property. Configure the following parameters:
	Default Query Condition Settings	
	Authorization Code	
	Derived Field	<ul style="list-style-type: none"> ◦ Search Item Configuration: Search items are displayed in the drop-down list only after they have been configured in Configure a search item. ◦ Default Query Condition Settings: The default condition used for a link query. If other properties are used as conditions for a query, this condition is also used by default. ◦ Authorization Code: After the authorization code function is enabled, only authorized users can access this property. ◦ Derived Property: After a property is set as a derived property, it can be generated automatically based on other properties. Configure the method in which the column is generated based on your needs.
	Delete	If a property is no longer used, you can click the Delete icon  to delete this property.
Sort order	The Move Up and Move Down arrows are used to adjust the order of properties displayed in Analytics Workbench.	

Category	Parameter	Description
Set Mappings Between Correlated Objects and Properties	Add Correlated Object	<p>Adds a mapping between an object and the event. One event must have two or more objects mapped to it.</p> <p>Click Add Correlated Object to add a correlated object, and configure the mapping between the event and the primary keys of the object you have added.</p>

- (Optional) After you have specified the required parameters, you can specify the optional parameters as needed.

The optional parameters are included in the **Advanced** and **Display Settings** areas. The optional parameters are described in [Optional event property parameters](#).

 **Note** Location Settings is currently not supported.

Optional event property parameters

Section	Parameter	Description
Advanced	Behavior Property	The properties based on which a behavior analysis is performed.
	Default Details Sorting Property	The property by which the details are sorted.
	Logical Relation of Authorized Properties	<p>The logical relationship between the authorization codes of properties in each record.</p> <ul style="list-style-type: none"> ◦ AND: The current record is visible only to the users who meet all authorization code conditions of the properties in this record. ◦ OR: The current record is visible to the users who meet any one authorization code condition of the properties in this record.
Display Settings	Enable Display	Indicates whether to show the event details.
	Group-by Properties	The property based on which events are aggregated. For example, aggregate Travel Events into a folder based on the Train Number property.

- After the configurations are complete, click **Save** in the upper-right corner. A success message is displayed after the modifications are saved.

An event is automatically enabled after its properties are saved.

8.7.2.3. Enable and disable an event

If you do not need to use a specific event during a specific period, you can disable this event, and this event can be re-enabled.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on

Analytics Workbench and Administration Console.

- To disable some events, you must first delete the dependency information of these events, including the mappings between the events and data tables and the objects referenced by the events.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. You can enable or disable an event by using one of the following methods.

Method	Operation
Operations on the Event Information page	In the left-side navigation pane, select an event to be enabled or disabled, and click the toggle switch to enable or disable the event.
Operations in Event Groups	<ol style="list-style-type: none">i. In the left-side navigation pane, select the event group that contains the event you want to enable or disable. Click the Event Information tab to go to the Event Information page.ii. Click the toggle switch to enable or disable the event. You can also select the events to be enabled or disabled, and click the Off button or the On button at the bottom of the page to disable or enable multiple events at a time.

8.7.2.4. View an event

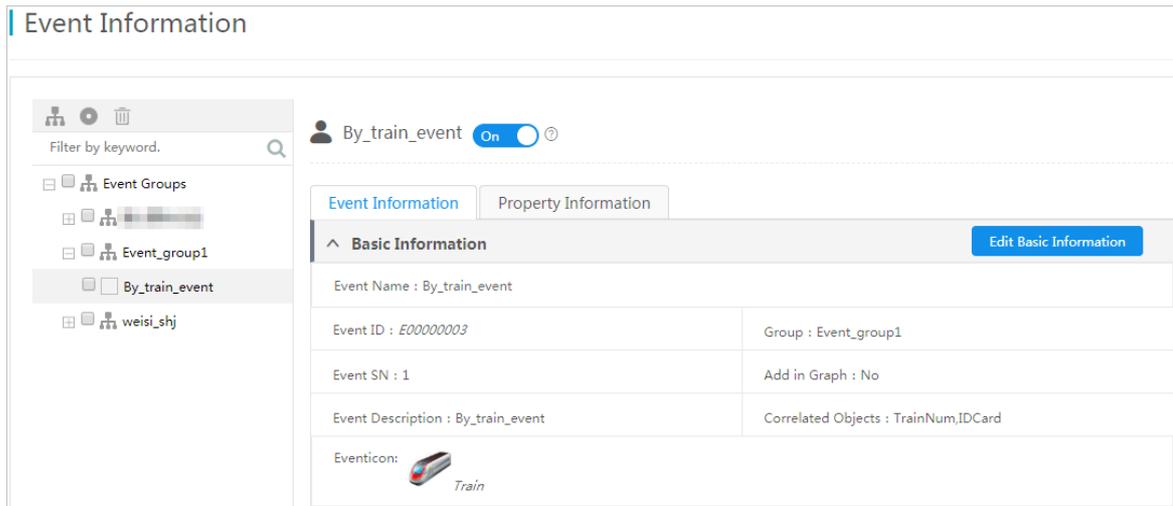
After you have configured an event, you can view the newly created event and all the events that have been created.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. In the left-side navigation pane, select an event and view the event details on the right side of the page.



Two tabs are displayed: The **Event Information** tab and the **Property Information** tab. The **Event Information** tab displays the basic information of the event. The **Property Information** tab displays the properties, correlated objects and property mappings, advanced settings, location settings, and display settings.

8.7.2.5. Modify an event

You can modify the basic information of the event you have created based on your needs.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. In the left-side navigation pane, select an event to be modified, and click **Edit Basic Information** to modify the parameters based on your requirements. The parameter configurations are described in [Modify parameter configurations of events](#). Modify parameter configurations of events

Parameter	Description
Event Name, Group, Event Description, and Event Icon	For more information about parameter descriptions, see Create an event .
Event SN	You can change the event sequence number based on your requirements.
Add in Graph	This parameter is not in use. You do not need to configure it.

4. After you have modified the parameters, click **Save**. A message is displayed, indicating that the modifications have been saved.

8.7.2.6. Delete an event

You can delete an event that is no longer used.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Click the **Events** tab in the top navigation bar. The **Event Information** page appears.
3. In the left-side navigation pane, select the events to be deleted, and click the **Delete** icon  on the top.
4. In the dialog box that appears, click **OK**. A message is displayed, indicating that the selected events have been deleted.

8.8. View the business graph

You can directly view all configured link models on the Business Graph page. Solid lines indicate first-degree links, and dotted lines indicate second-degree or multiple-degree links.

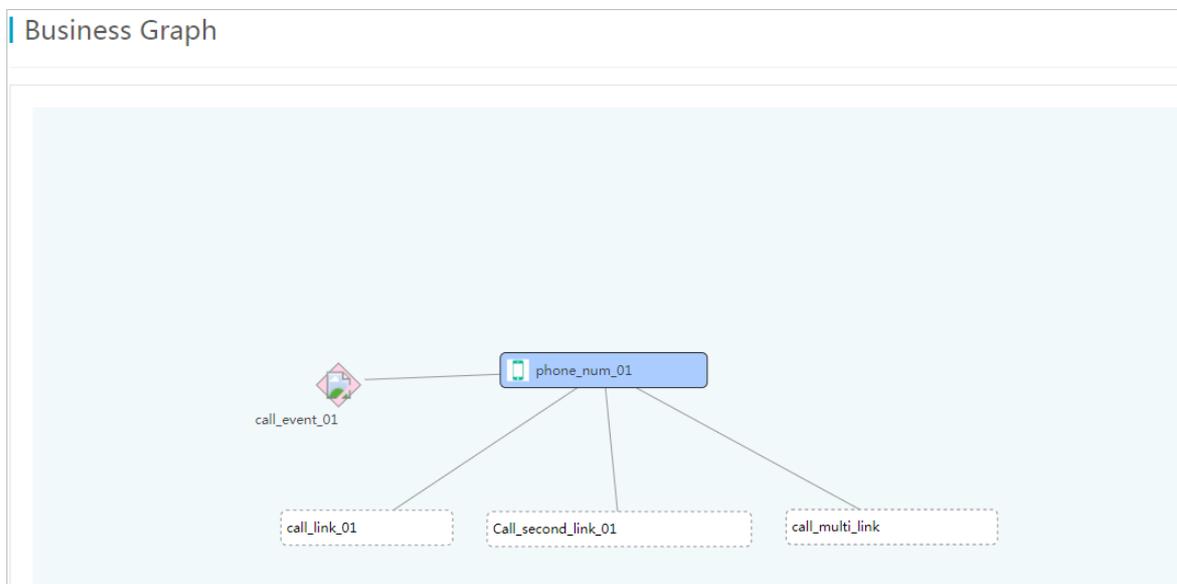
Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, click **Business Graph**.

On the **Business Graph** page, you can view the created link models between objects, links, and events.



8.9. Advanced configurations

8.9.1. Manage a system model

The Import Models page allows you to configure system models that are used to import data to Analytics Workbench.

Prerequisites

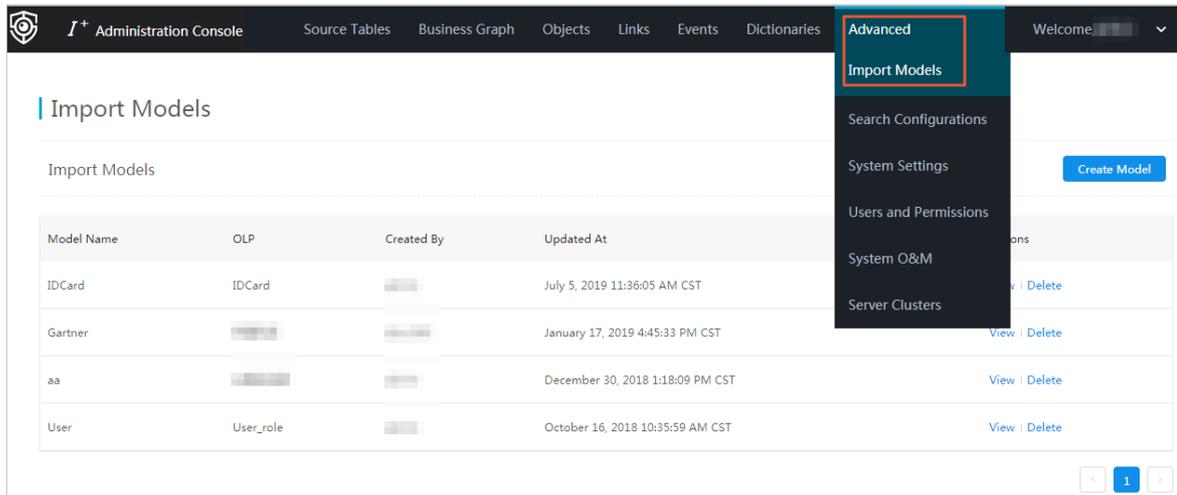
Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Context

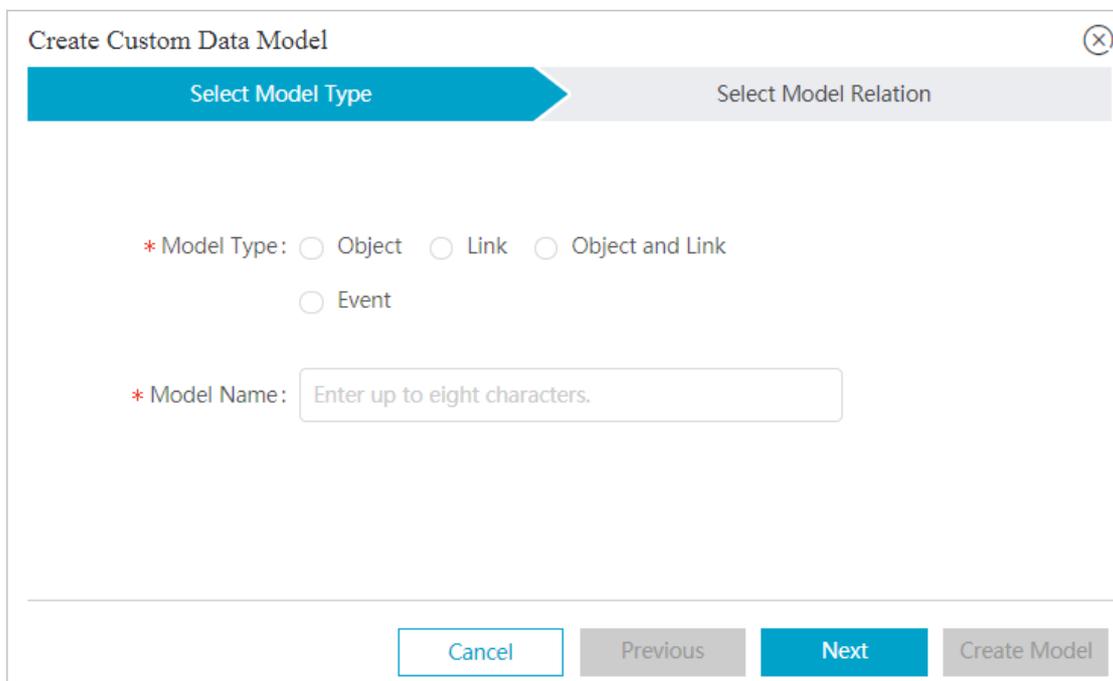
You can create, view, and delete models on the Import Models page. This topic provides examples to help you learn more about these operations.

Create a model

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, choose **Advanced > Import Models**.

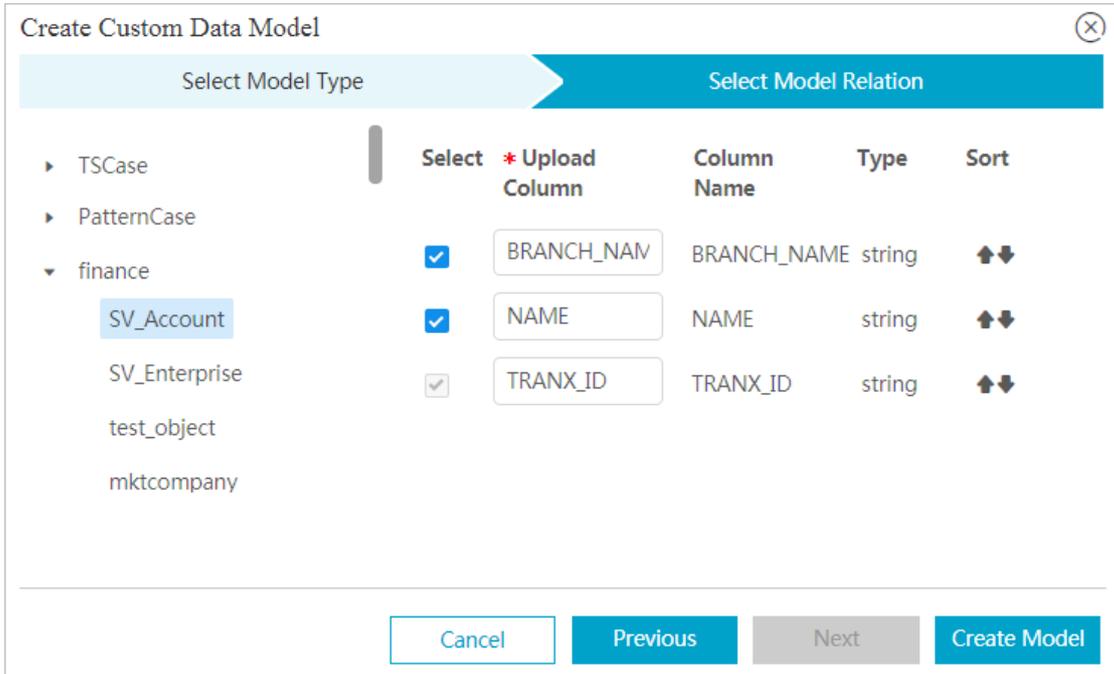


3. On the **Import Models** page, click **Create Model** in the upper-right corner of the page.
4. In the **Create Custom Data Model** dialog box, specify the **Model Type** and the **Model Name**.



5. Click **Next**.
6. Set the model relation in the **Create Custom Data Model** dialog box.

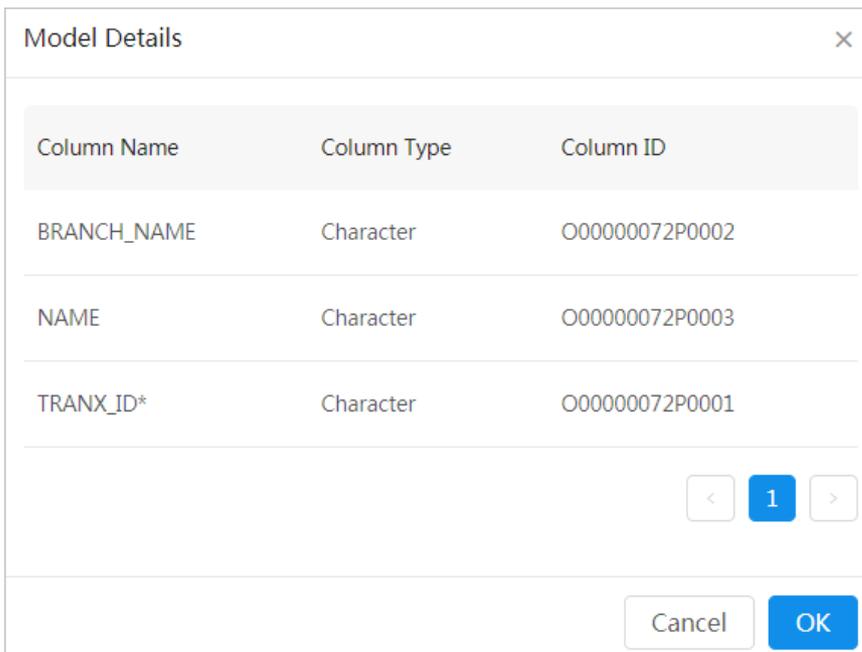
Select an object, link, or event on the left side, and select the model columns on the right side based on the data to be imported. The columns of the primary key type are selected by default and cannot be operated.



7. After you have configured the preceding parameters, click **Create Model**.

View a model

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, choose **Advanced > Import Models**.
3. On the **Import Models** page, select a model and click **View** to view the details of the model in the **Model Details** dialog box that appears.



Delete a model

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > Import Models.**
3. On the **Import Models** page, select a model and click **Delete.**
4. In the dialog box that appears, click **OK** to delete the model.

8.9.2. Configure a search item

You can configure search items to set the fields to be searched for in Analytics Workbench. After a search item is configured, it must be correlated with an object, link, or event property to take effect.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- Before you delete a search item, you must first delete the links between the search item and the correlated properties of objects, links, and events.

Context

This topic describes how to add, modify, and delete search items, and how to associate search items with the properties of objects, links, or events.

Add a search item

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > Search Configurations.**
3. Click **Add Item** in the upper-left corner of the page that appears.

The parameters are described in [Search item configuration parameters.](#)

* Search Item Name	Search Item Type	Advanced Correlated Items	Show in Main Search Box
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input checked="" type="checkbox"/>
<input type="text" value="Name"/>	<input type="text" value="String Like"/>	<input type="text" value="IDcard_num"/>	<input checked="" type="checkbox"/>
<input type="text" value="IDcard_num"/>	<input type="text" value="String Like"/>	<input type="text" value="Name"/>	<input checked="" type="checkbox"/>
<input type="text" value="BRANCH_NAM"/>	<input type="text" value="String Like"/>	<input type="text"/>	<input type="checkbox"/>
<input type="text" value="Account_name"/>	<input type="text" value="String Like"/>	<input type="text" value="BRANCH_NAME"/>	<input checked="" type="checkbox"/>
<input type="text" value="Enterprise_nan"/>	<input type="text" value="String Like"/>	<input type="text"/>	<input checked="" type="checkbox"/>

Search item configuration parameters

Parameter	Description
Search Item Name	Search Item Name is customized by the user. We recommend that you set the name according to the properties of the objects, links, or events that need to be correlated. For example, a mobile phone number, an ID number, or a person's name.

Parameter	Description
Search Item Type	The Search Item Type is used to set the data type that is supported by this search item. The Search Item Type must be consistent with the Query Type of the property of the object, link, or event to be correlated with.
Advanced Correlated Items	Advanced Correlated Items are used to group multiple search terms to search for data. You can select multiple configured search terms.
Show in Main Search Box	Sets whether the search item is displayed in the Search page of Analytics Workbench.

4. (Optional) Click **Add Item** to add multiple search items as needed.
5. Click **Save**.

After a search item is added, you must correlate the search item to the properties of an object, link, or event. For more information, see [Correlate a search item to properties](#).

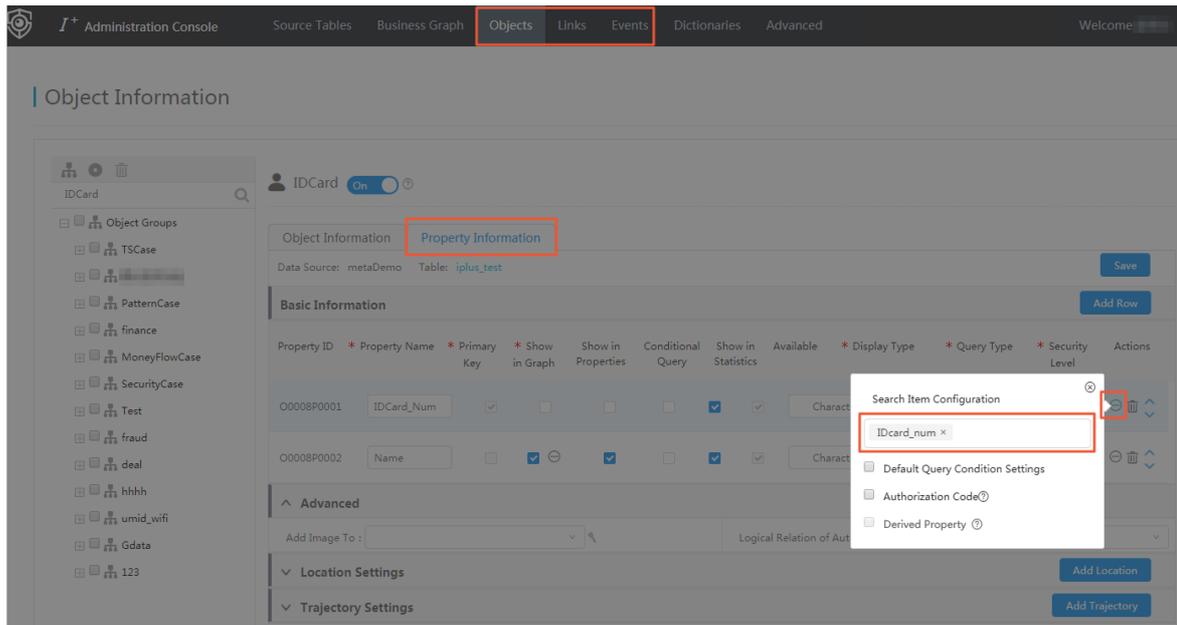
Correlate a search item to properties

The configuration entries are as follows:

- **Objects:** Administration Console > Objects > selected the object to be correlated to > Property Information > Basic Information > more configurations.
- **Links:** Administration Console > Links > select the link to be correlated to > Property Information > Basic Information > more configurations.
- **Events:** Administration Console > Events > select the event to be correlated to > Property Information > Basic Information > more configurations.

This topic describes how to correlate a search item with the properties of an object as an example. You can correlate a search item with the properties of a link or event by using similar methods.

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, click **Objects**.
3. In the left-side navigation pane of the **Object Information** page, click the name of the object to be configured and then click the **Property Information** tab on the right side.
4. In the **Basic Information** area, click the More icon (⊕) next to the property to be correlated, and then select the configured search items in **Search Item Configuration**.



5. After you have configured these parameters, click the  icon to close the configuration box.
6. Click **Save** to save the configurations.

Modify a search item

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > Search Configurations**.
3. Modify the parameters of a search item as needed.
4. Click **Save**.

Delete a search item

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > Search Configurations**.
3. Click the  icon to delete the corresponding search item.
4. Click **Save**.

8.9.3. System settings

8.9.3.1. Configure components

You can configure the functional components to enable or disable functions in Analytics Workbench and Administration Console and set basic information of Analytics Workbench. When a functional component is disabled, this function will not be displayed in the operation area.

Prerequisites

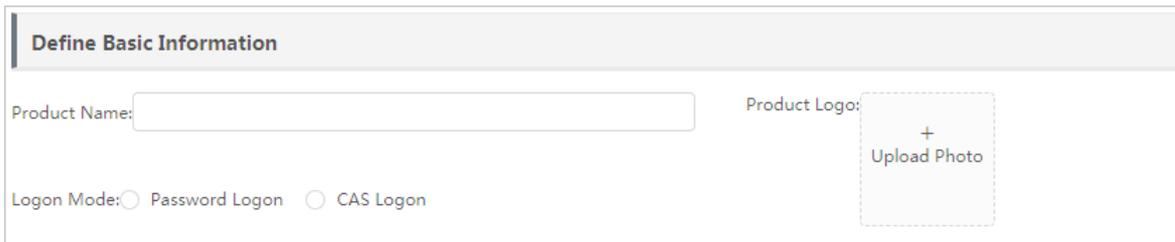
Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Context

The functional component settings take effect globally. Enable or disable components as needed with caution.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, choose **Advanced > System Settings > Functional Components**.
3. Define basic information: Set the **Product Name**, **Product Logo**, and **Logon Mode** parameters for Graph Analytics.

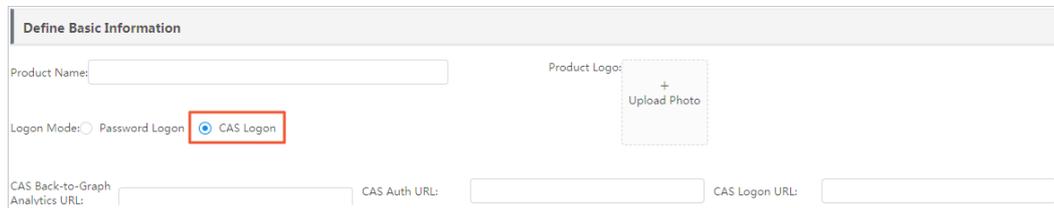


Note If you do not set the **Product Name** parameter and the **Product Logo** parameter, the default name and default Logo image are used.

If the CAS (Central Authentication Server) server is deployed in your environment, you can select **CAS logon** as the logon mode, as shown in [CAS logon](#).

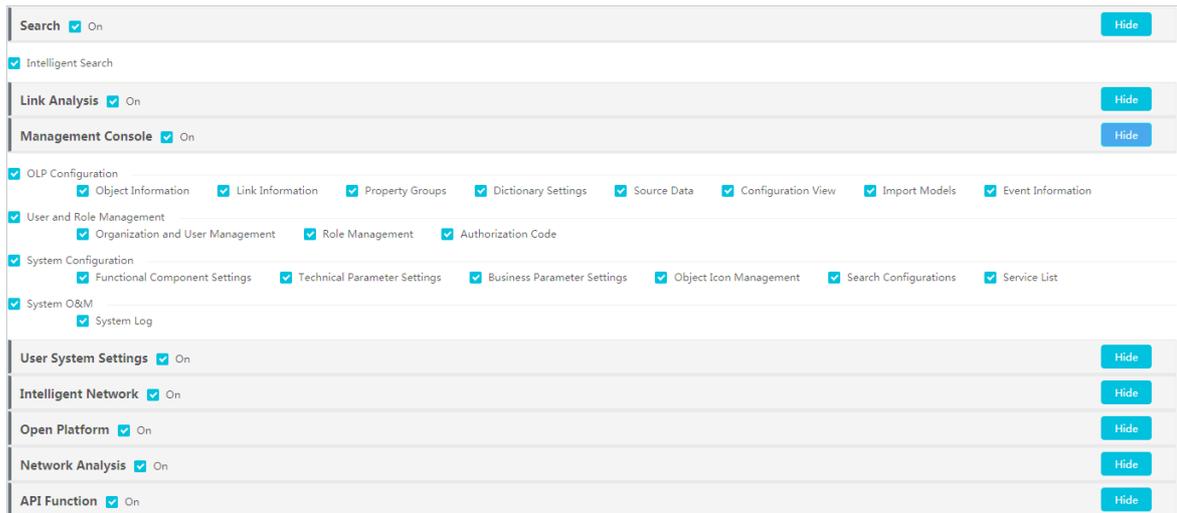
Note CAS is a single sign-on protocol. If you select **CAS logon** as the logon mode, you only need to log on once to access all trusted application systems.

CAS logon



4. Enable and disable functional components: Select the modules to be enabled and cancel the modules to be disabled.

If you select or clear a parent component, the child components will be automatically selected or cleared.



5. Click **Save**.

8.9.3.2. Technical parameters

8.9.3.2.1. Path analysis settings

In Graph Analytics, you can set the highest link degree of path analysis. Also, you can set whether to calculate only the nodes that are of the same type as the selected nodes.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

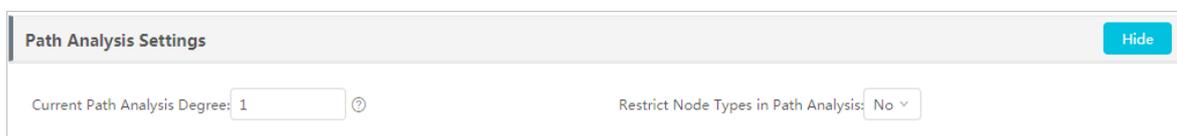
Context

Typically, the default values are used for path analysis settings. Exercise caution when you modify the settings.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > System Settings > Technical Parameters**.
3. In the **Path Analysis Settings** area, you can set the parameters based on your requirements.

The parameters are described in [Path analysis configuration parameters](#).



Path analysis configuration parameters

Parameter	Description
-----------	-------------

Parameter	Description
Current Path Analysis Degree	<p>The highest link degree supported by path analysis. The default value is 2 and the maximum value is 3.</p> <p>If the Current Path Analysis Degree parameter is set to <i>N</i>, only links of degree <i>N</i> or lower degrees will be analyzed. Links of a degree higher than <i>N</i> will not be analyzed.</p>
Restrict Node Types in Path Analysis	<p>Valid values:</p> <ul style="list-style-type: none">◦ Yes: Only nodes of the same type as the selected node are calculated.◦ No: All types of nodes are calculated. <p>The default value is No.</p>

4. Click **Save** in the upper-right corner.

8.9.3.2.2. Quick extension settings

In Graph Analytics, you can set the highest link degree for quick extension.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Context

Typically, the default values are used for quick extension settings. Exercise caution when you modify the settings.

 **Note** If the degree is set too high, the running performance of the system will be affected.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > System Settings > Technical Parameters**.
3. In the **Quick Extension Settings** area, you can set the **Extension Degree** parameter based on your requirements.



Extended Degree: The highest link degree for quick extension. The default value is 2. If the **Extended Degree** parameter is set to *N*, only links of degree *N* or lower degrees will be analyzed. Links of a degree higher than *N* will not be analyzed.

4. Click **Save** in the upper-right corner.

8.9.3.2.3. Maximum node settings

In Graph Analytics, you can set the maximum number of nodes to be queried at the same time when you perform an analysis.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Context

Typically, the default value is used for the maximum number of nodes to be queried. Exercise caution when you modify the number of nodes.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > System Settings > Technical Parameters.**
3. In the **Maximum Node Settings** area, you can configure the parameters based on your requirements.

4. After you have completed the configurations, click **Save** in the upper-right corner.

8.9.3.3. Business parameters

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Choose **Advanced > System Settings > Business Parameters** from the top navigation bar. The Business Parameters page appears.

8.9.3.3.1. Add double-click link settings

In Analytics Workbench, you can double-click an object to query the relationship between objects. In Administration Console, you can custom the relationships of an object to be queried when you double-click the object. If no custom configuration is set, all the first-degree relationships of the object are queried by default when you double-click the object.

Prerequisites

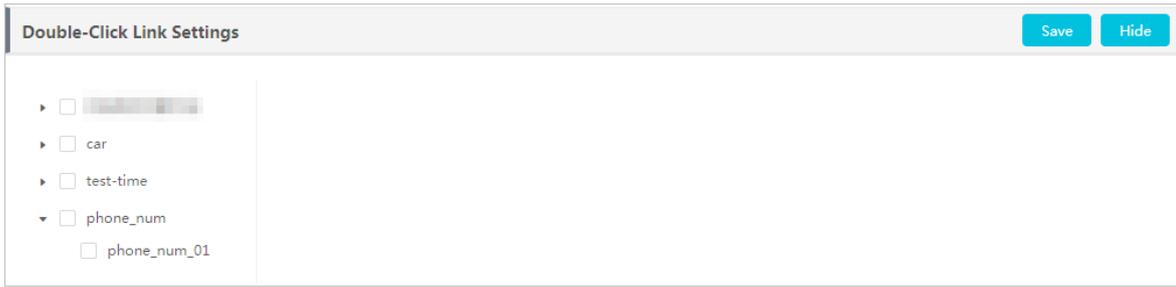
- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- The object has been created and referenced by the link.

Context

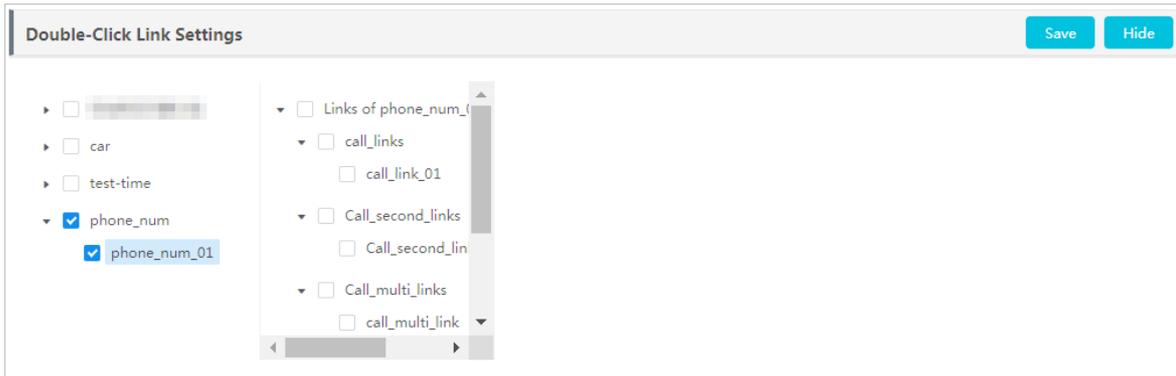
The double-click link settings take effect globally. Exercise caution when you configure the settings.

Procedure

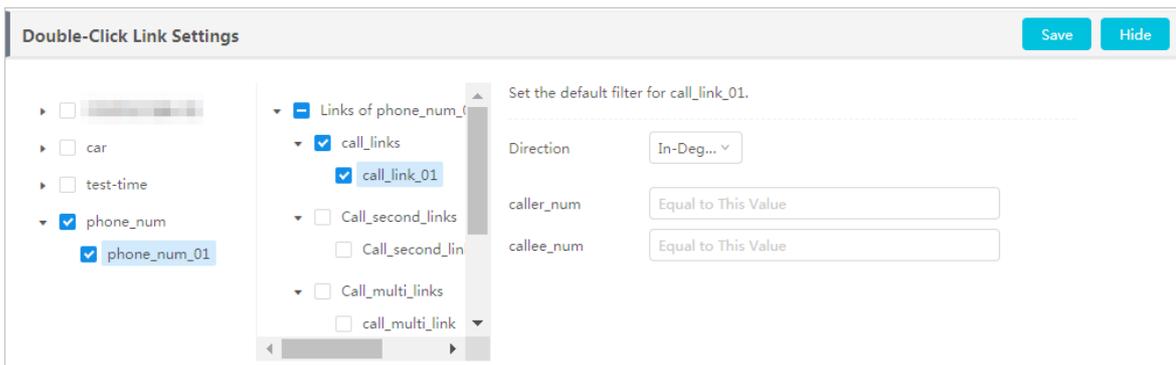
1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > System Settings > Business Parameters.**
3. In the **Double-Click Link Settings** area, select the objects to be configured in the left-side list.



4. In the list that is displayed on the right side of the area, select the links to be queried.



5. In the displayed area, set the filter conditions for querying the link.



6. (Optional) If you need to query other links by double-clicking the object, you can continue to select other links and set the filter conditions.

7. Click **Save** to complete the double-click link settings of the object.

8.9.3.3.2. Double-click-disabled object settings

In Analytics Workbench, you can double-click a node to query the relationship network of the node. Administration Console allows you to enable or disable double-clicking on a specified object. By default, double-clicking is supported for all objects.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have created an object.

Context

The double-click-disabled object settings take effect globally. Exercise caution when you configure the settings. We recommend that you disable the double-click operations on objects that generate large amounts of data.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, choose **Advanced > System Settings > Business Parameters**.
3. In the **Double-Click-Disabled Object Settings** area, click the **Double-Click-Disabled Objects** drop-down list, and select the objects.



4. Click **Save**.

8.9.3.3.3. Object grouping settings

Graph Analytics allows you to set the grouping conditions, so that objects or links of the same type can be grouped automatically when their quantity reaches the threshold value. In a complex graph analytics, we recommend that you set reasonable grouping conditions to keep the analysis graph concise and clear.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have created an object.

Context

The **Object Grouping Threshold** and **Object Grouping Degree** parameters are described as follows:

- **Object Grouping Threshold:** If the number of object nodes of the same type that are connected to the same node exceeds the threshold, the connected nodes are grouped into one folder.
- **Object Grouping Degree:** If more than one node has the link of the specified degree with the same nodes, these nodes will be grouped into a folder. For example, set **Object Grouping Degree** to 2. If both node A and node B have second-degree links with nodes C and D, node A and node B will be grouped into one folder.

The value of **Object Grouping Threshold** defaults to 0, and the value of **Object Grouping Degree** defaults to 1. The default value indicates that no objects will be grouped automatically.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, choose **Advanced > System Settings > Business Parameters**.
3. In the **Object Grouping Settings** area, specify **Object Grouping Threshold** and **Object Grouping Degree** based on your requirements.

Object Name	Object Grouping Threshold	Object Grouping Degree
Source_account	0	1
mktclient	0	1
identity_card	0	1
wifi_fin	0	1

4. Click **Save**.

8.9.3.3.4. Configure lineage analysis

Before you perform a lineage analysis on a specified object node in Analytics Workbench, you need to configure the business parameters for this type of object in Administration Console.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- The object has been created and referenced by a link.

Context

Lineage analysis extends a specific business link to multiple degrees. The specific link typically refers to the lineage link and the same residence number.

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. In the top navigation bar, choose **Advanced > System Settings > Business Parameters**.
3. In the **Lineage Analysis Settings** area, click **Add** to add a new setting.
4. Select the object to be analyzed based on your requirements, and then select the object-related link to be analyzed.

Lineage Analysis Settings		Save	Add	Hide
Select Object:	phone_num_01	Select Object-Related Links:	call_link_01	

5. Click **Save**.

8.9.3.3.5. Intimacy measurement settings

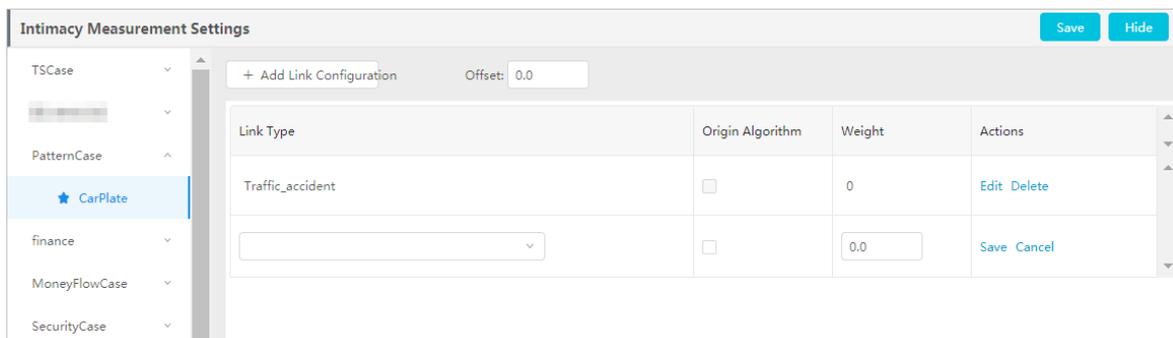
You can set the weight for the relationship between specified objects, so that you can directly view the intimacy between these objects in the analysis result.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- An object has been created and has been referenced by a link.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, choose **Advanced > System Settings > Business Parameters**.
3. On the left side of the **Intimacy Measurement Settings** area, select an object and then click **Add Link Configuration** on the right side of the area.
4. Select a link type in the **Link Type** drop-down list, and then set the intimacy weight and other parameters.



The star icon in front of an object indicates that the object has configured intimacy measurement settings, as shown in ID Card in the figure above.

5. After you have configured these parameters, click **Save**.
6. (Optional) To add intimacy measurement settings for another **Link Type**, click **Add Link Configuration**.
To add intimacy measurement settings for another object, select the object in the left-side navigation pane, and then click **Add Link Configuration**.
7. After you have configured the intimacy measurement settings for all objects, click **Save** on the right side of the **Intimacy Measurement Settings** section.

8.9.3.3.6. Redirect URL settings

When you need to redirect to an external system from Graph Analytics, you can configure the URLs in External System Redirect URL Settings.

8.9.3.4. Object icons

8.9.3.4.1. Upload an object icon

You can upload a local object avatar to the icon library in Graph Analytics.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- Make sure that you have prepared an avatar image in the PNG format. The recommended size is 32px * 32px. We recommend that you limit the size to 320px * 320px.

Procedure

1. Log on to Administration Console of Graph Analytics.
2. In the top navigation bar, choose **Advanced > System Settings > Object Icons**.
3. On the **Object Icons** page, click **Upload Icon**.

4. In the **Upload Icon** dialog box that appears, click the upload area to upload a local icon, and then specify the **Name** column.
5. Click **OK** to upload the object icon.

8.9.3.4.2. Modify an object icon

You can modify the avatar image or the avatar name based on your requirements.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- Make sure that you have prepared an avatar image in the PNG format. The recommended size is 32px * 32px. We recommend that you limit the size to 320px * 320px.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top menu bar, choose **Advanced > System Settings > Object Icons**.
3. On the **Object Icons** page, move your mouse pointer over the icon, and click the **Modify icon** ()
4. In the **Edit** dialog box that appears, modify **Name** or upload a new avatar image.
5. Click **OK**.

8.9.3.4.3. Delete an object icon

You can delete the object icons that are no longer used.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- Before you delete an object icon, make sure that this icon is not used by another object.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. In the top navigation bar, choose **Advanced > System Settings > Object Icons**.
3. On the **Object Icons** page, move your mouse pointer over the icon, and click the **Delete icon** ()
4. In the dialog box that appears, click **OK** to delete the object icon.

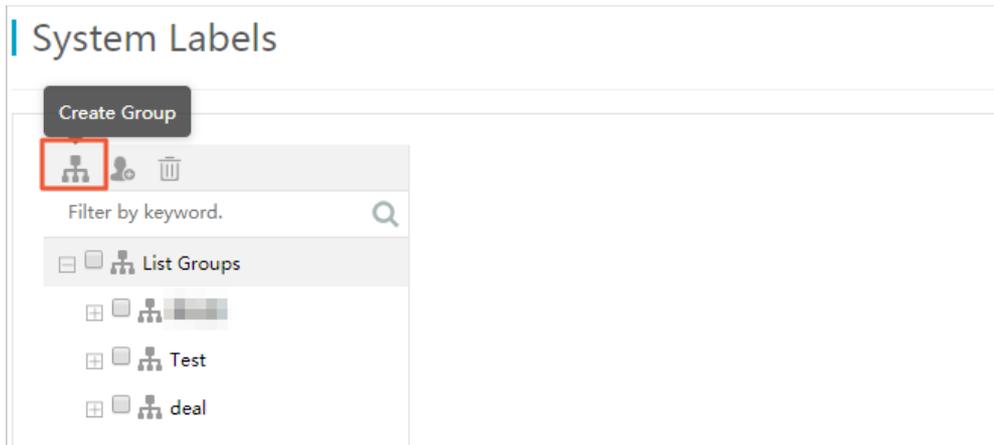
8.9.4. System labels

8.9.4.1. Create a group

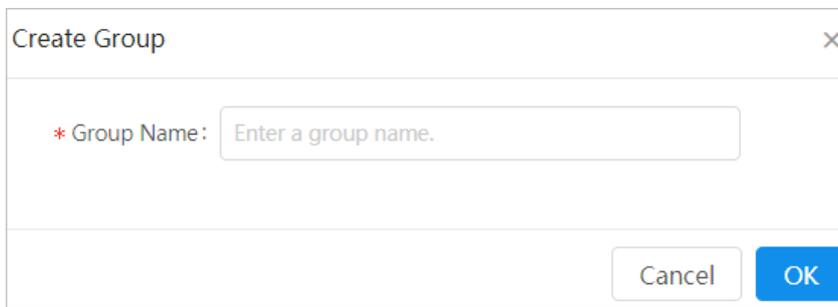
Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Choose **Advanced > Users and Permissions > System Labels** from the top navigation bar.
3. On the **System Labels** page, click the **Create Group** icon, as shown in [Create a group](#).

Create a group



4. In the Create Group dialog box that appears, enter a group name, as shown in [Enter a group name](#).



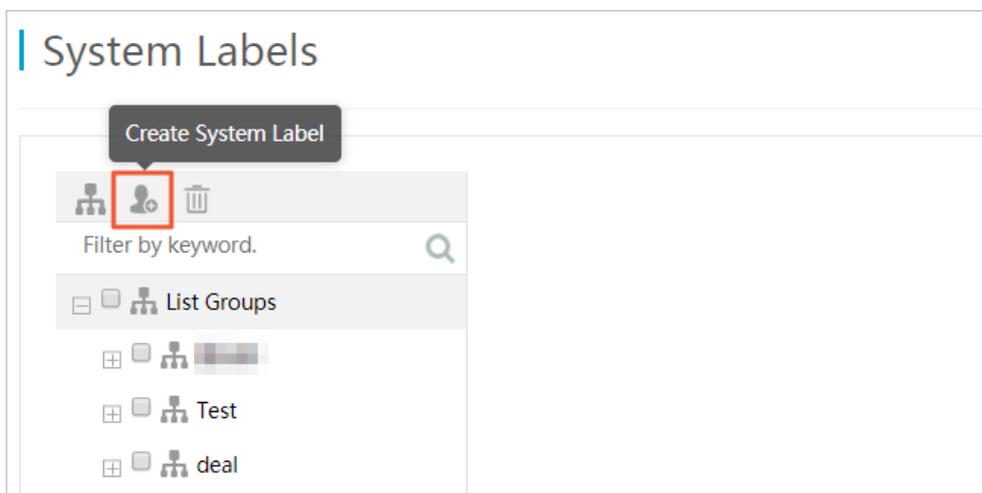
5. Click OK.

8.9.4.2. Create a system label

Procedure

1. [Log on to Administration Console of Graph Analytics](#).
2. Choose **Advanced > Users and Permissions > System Labels** from the top navigation bar.
3. On the System Labels page, click the **Create System Label** icon, as shown in [Create a system label](#).

Create a system label



4. In the **Configure System Label** dialog box, configure the required parameters, as shown in [Configure a](#)

system label.

Configure a system label

Configure System Label

* System Label Na... Enter a system label name.

* Group: Select a group

Rule Definition: Select a rule definition.

Label Color: Select a label color.

* Object Type: Select the object type.

* Data Sources: Select a data source.

* Table: Select a table.

Object Type Filter-... Select an object type filter-by column.

Cancel OK

The system label parameters are described as follows:

- o System Label Name: the alias of the system label.
- o Group: the group to which the system label belongs.
- o Rule Definition: The processing rule that is applied when an object matches the system label. Value options include Not Display Object, Not Extend Object, Not Display Properties, Disable Property Statistics, and Disable Behavior Display. If you have selected Not Display Object, the other four options and the Label Color parameter become unavailable.
- o Label Color: the display color of the system label.
- o Object Type: all available objects are displayed in the drop-down list. Choose an object as needed.
After you select the object type, a primary key parameter will appear in the dialog box. Select a column corresponding to the primary key property as needed.
- o Data Source: the data source where the system label belongs.
- o Table: the table where the system label belongs.
- o Object Type Filter by Column, Object Type Filter Operator, and Object Type Filter Value: Optional. You can use these parameters to filter system labels. If you have specified any one of these parameters, the other two parameters are required.

5. Click OK.

8.9.4.3. Modify a system label

Procedure

1. Log on to Administration Console of Graph Analytics.
2. Choose **Advanced > Users and Permissions > System Labels** from the top navigation bar.
3. On the System Labels page, click an object from the left-side navigation pane.
4. In the details area on the right side of the page, select a system label, and click **Edit**, as shown in [Modify a system label](#).

Modify a system label



5. Modify the parameters as needed. The object type cannot be modified.
6. Click **OK**.

8.9.4.4. Delete a system label

Procedure

1. Log on to Administration Console of Graph Analytics.
2. Choose **Advanced > Users and Permissions > System Labels** from the top navigation bar.
3. Click **Delete** next to a system label, or select one or more system labels and then click **Delete** in the lower-left corner, as shown in [Delete a system label](#).

Delete a system label



8.9.5. System operations and maintenance

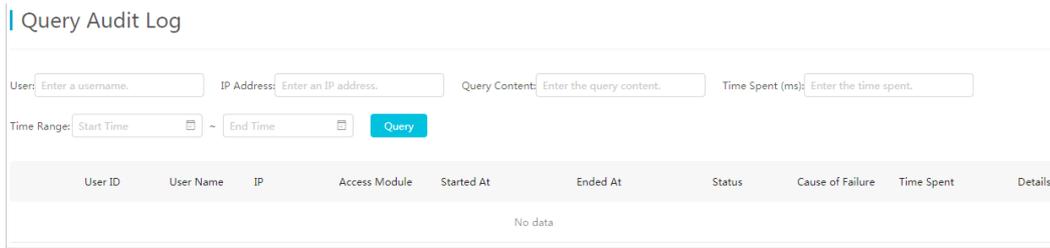
8.9.5.1. Audit logs

Procedure

1. Log on to Administration Console of Graph Analytics.
2. Choose **Advanced > System O&M > Audit Log** from the top navigation bar. The Query Audit Log page appears.
3. Set the search filters, and then click **Query**, as shown in [Log query](#).

If you click **Query** without setting any filters, all data will be returned.

Log query



8.9.6. View server clusters

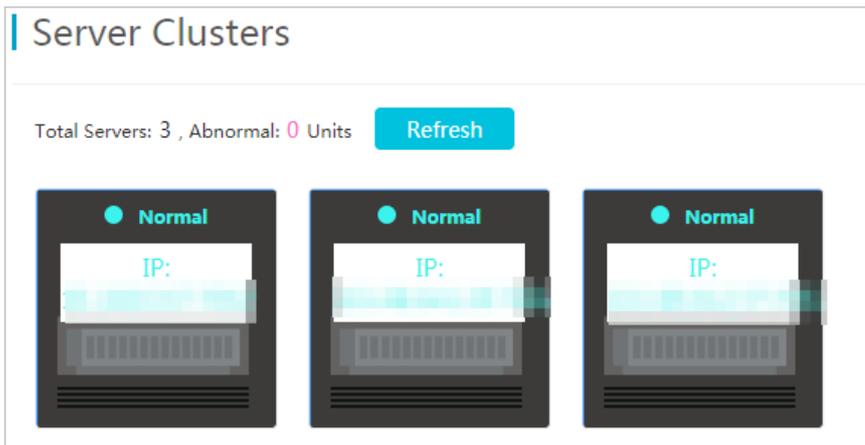
In Administration Console, you can view the information of all servers in a cluster, including the running status, server exceptions, the number of servers, IP addresses, and port numbers.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Administration Console of Graph Analytics.](#)
2. Choose **Advanced > Server Clusters** to go to the **Server Clusters** page.



In Administration Console, you can view the information of all servers in a cluster, including the running status, the number of servers and server exceptions, IP addresses, and port numbers.

3. If you have stayed on the current page for too long, you can click **Refresh** to view the latest information.

8.10. Import data

8.10.1. Model list

8.10.1.1. Model overview

A model is a template used to import data to Graph Analytics. You can use models to import individual or small amounts of data to Graph Analytics.

Models include custom models and system models:

- Custom models are directly created by users on Analytics Workbench. Only the creator can view, download, modify, and delete the custom models.
- System models are created by the administrator in Administration Console. All users can view and download

system models on Analytics Workbench. However, system models cannot be modified or deleted. To delete a system model, you must log on to Administration Console.

8.10.1.2. View models

In Analytics Workbench, you can view information about the existing data models, including the model type, mapped OLEP, properties, and the property type. This helps you understand the existing models at any time and identify the data model that matches the data to be imported.

Prerequisites

All users can view the system models. However, to view a custom model, you must have the account and password of the user that created the model.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the  icon in the upper-right corner of the page, and click the **Model List** tab. On the **Model List** page, you can view the creation time, model type, and mapped OLEP for all models.

 **Note** Analytics Workbench supports custom and system models. Custom models are created by users on Analytics Workbench. For more information, see [Create models](#). System models are configured in Administration Console. For more information, see [Manage a system model](#).

Data List		Model List				Create Model
Model Name	Created At	Type	Mapped OLEP	Type	Actions	
 calllink	January 28, 2019 2:24:49 PM GST	Link	call_link_01	Custom	  	
phone_num	January 22, 2019 5:17:00 PM GST	Object	phone_num_01	System	 	

3. Click the  icon next to the data model that you want to view, and view the property information about the data model in the dialog box that appears.

Model : calllink ✕

Property Name	Required	Property Type
callee_num*	Yes	Character (string)
caller_num*	Yes	Character (string)

Cancel
Download Template File

If the data model meets the requirements for your data import, you can download this model and use this model as a template to sort the data to be imported.

4. (Optional) Click **Download Template File** to download the model file in the XLSX format.

8.10.1.3. Create models

The model list defines a column format for the uploaded file. The columns of the uploaded file will be mapped to the corresponding properties, including the object property, link property, object and link property, and event property. If the existing models cannot meet the requirements of the data to be imported, you must create a data model based on the target data.

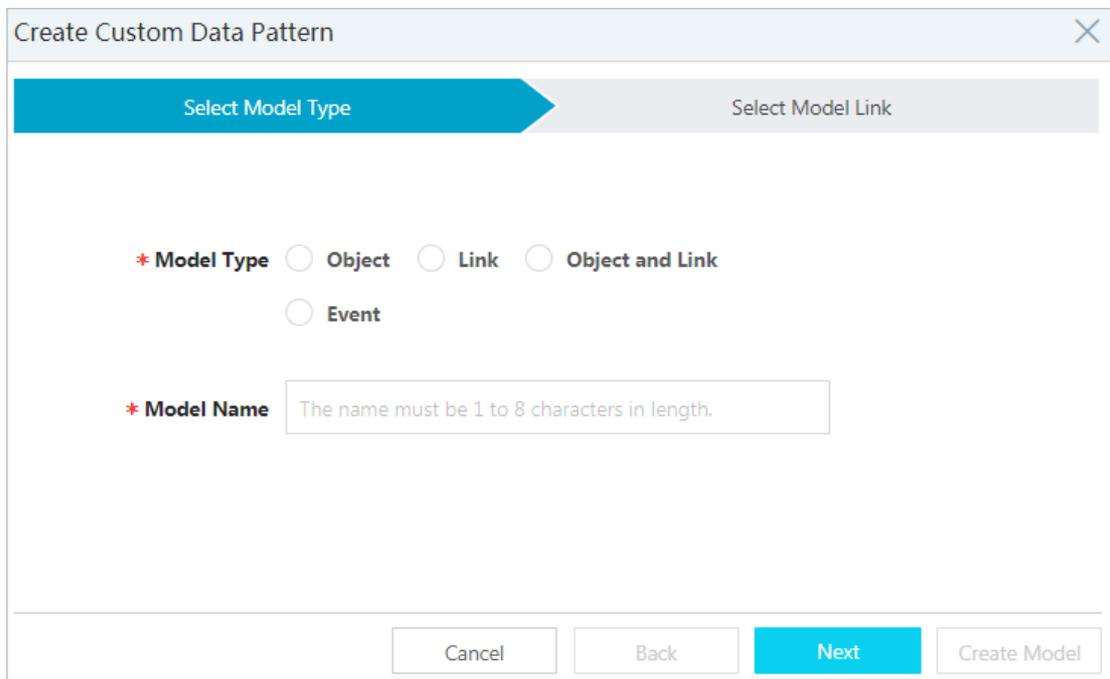
Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click the  icon in the upper-right corner of the page, and then click the **Model List** tab. The **Model List** page appears.
3. Click **Create Model** in the upper-right corner of the page, and set the parameters in the **Create Custom Data Model** dialog box that appears.

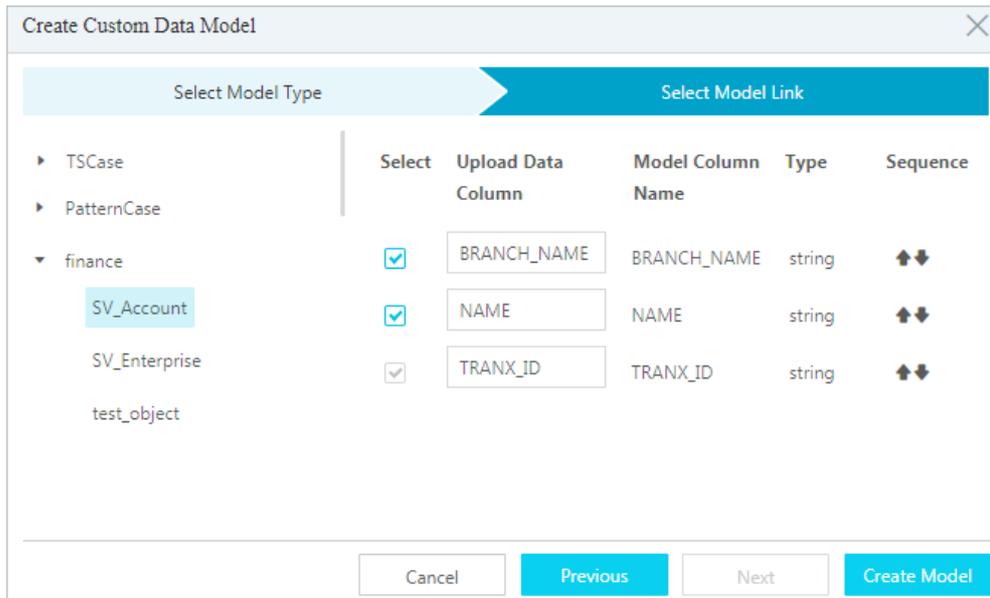
For more information about the parameter settings, see [Parameter configurations](#).



Parameter configurations

Parameter	Description
Model Type	You can create multiple model types, including Object , Link , Object and Link , and Event . Specify a model type as needed.
Model Name	Enter a model name. We recommend that you enter a name that is easy to understand.

4. Click **Next**, select an object, link, or event, select the required columns, and then set the name of **Upload Data Column**.



5. Click **Create Model**. A success message is displayed, indicating that the model has been saved.

8.10.1.4. Modify model names

If a model name is obscure or does not match the model content, you can modify the model name.

Prerequisites

Make sure that you have obtained the account and password of the user that created the model that you want to modify.

Note In Graph Analytics, you can only modify the names of user-defined models. You are not allowed to modify the system models.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click the icon in the upper-right corner of the page, and click the **Model List** tab. The **Model List** page appears.
3. Click the icon in the front of the model that you want to edit, rename the model, and then click **OK**.

8.10.1.5. Download a model

Before you import data, you must download a compatible model, which is an .xlsx file, and use this model as a template to sort the data to be imported.

Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click the icon in the upper-right corner of the page, and click the **Model List** tab. On the **Model List** page, you can view the creation time, model type, and mapped OLEP for all models.

3. Click the  icon next to the model that you want to download, and save the model as prompted.

What's next

Use this model as a template to organize the data to be imported, and then import the collected data to Graph Analytics. For more information about the detailed operations, see [Import data](#).

8.10.1.6. Delete a model

You can delete the custom models that are no longer used.

Prerequisites

Make sure that you have obtained the account and password of the user that created the model you want to delete.

 **Note** Graph Analytics only supports deleting user-defined models. To delete a system model, you need to log on to Administration Console. For more information, see [Manage a system model](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Click the  icon in the upper-right corner of the page, and click the **Model List** tab. On the **Model List** page, you can view the creation time, model type, and mapped OLEP for all models.
3. Click the  icon next to the model that you want to delete.

8.10.2. Import data

Analytics Workbench supports importing data in the csv, txt, xls, or xlsx format. You can analyze small amounts of data or individual pieces of data that are missing from the data source.

Prerequisites

- The import data source is added when you configure the data source [Create data sources](#).
- You have obtained an account and a password with the **Import Data** permissions.
- You have created a model matched to the data to be imported. For more information, see [Create models](#).
- You have organized the data to be imported according to the template file, or sorted the data in the format required by the template, such as the csv, txt, xls, or xlsx format.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click the  icon in the upper-right corner. In the dialog box that appears, select the file to be imported.
3. Click **Upload**, and set the read format of the file according to the data content to be imported.

The parameter configurations are described in [Parameter configurations and descriptions](#).

After the file is imported, you can see the data preview of the file in the dialog box. Only the first 10 lines of the file content and the total number of lines are displayed.

Import Data - 2. Select Model
✕

Select a file to upload: Upload

Data Preview Total: 3 Items

A0	A1	A2	A3
32	John	2010-10-12 00:00:00	male
23	Lili	1997-09-10 00:00:00	female
13	Tom	2010-10-12 00:00:00	male

Field Separator: Comma (,) Semicolon (;) Tab

Pre-Filter Row Table Head

Select Model Object Link Object and Link Event

System Model phone_num fsd ygdf dd hhh

Custom Model PhoneNum ID_Card

Create Model

Cancel
Next

Parameter configurations and descriptions

Parameter	Description
Column Separator	Sets the internal column separator for each line of the content.
Encoding Method	Specifies the character encoding for the file content.
Select Model	After you have selected the model type corresponding to the uploaded data, you can see the currently available system models in the System Model tab and custom models in the Custom Model tab. If you do not have a model that matches the data to be imported, you can click Create Model to create a new model in real time. For more information, see Create models .

- After you have configured the parameters, click **Next** to set the data name and the mapping relationship between the columns and model properties of the data to be imported.

Import Data - 3. Configure Data
✕

Selected Model - ID_Card Upload Data - IDcard.xlsx

<input type="text" value="IdentityCard*"/>	<input type="text" value="Name"/>	<input type="text" value="Birthday"/>	<input type="text" value="Sex"/>
3	John	2010-10-12 00:00:00	male
2	Lili	1997-09-10 00:00:00	female
1	Tom	2010-10-12 00:00:00	male

Data Name:

Back
Submit Data Import

- After you have configured the parameters, click **Submit**.

8.10.3. Data list

8.10.3.1. View data

Analytics Workbench allows you to view the imported data at any time, so that you can better understand the data.

Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the  icon in the upper-right corner of the page to go to the **Data List** tab page. You can view all data that has been imported.

The Data List page displays information about all imported data, including the data name, model name, upload time, OLEP type, mapped OLEP, and the import status.

Data List		Model List				
Data Name	Model Name	Upload Time	OLEP Type	Mapped OLEP	Import Status	Actions
 IDCard	IDCard	July 15, 2019 11:12:39 AM CST	Object	IDCard	Success	  

3. Click the  icon next to a data model that you want to view, and view the property information about this data in the dialog box that appears.

8.10.3.2. Edit a data name

If a data name is obscure or does not match the data content, you can modify the data name.

Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the  icon in the upper-right corner of the page to go to the **Data List** page.

Data List		Model List				
Data Name	Model Name	Upload Time	OLEP Type	Mapped OLEP	Import Status	Actions
 IDCard	IDCard	July 15, 2019 11:12:39 AM CST	Object	IDCard	Success	  

3. Click the  icon in front of the data name, rename the data, and then click **OK**.

8.10.3.3. Import data to Graph

You can use this feature to import a data file to Graph to analyze the data quickly.

Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the  icon in the upper-right corner of the page to go to the **Data List** page.

Data List		Model List				
Data Name	Model Name	Upload Time	OLP Type	Mapped OLP	Import Status	Actions
IDCard	IDCard	July 15, 2019 11:12:39 AM CST	Object	IDCard	Success	  

3. Click the  icon next to the specified data file to present the data in Graph.

8.10.3.4. Delete data

You can delete unnecessary data.

Prerequisites

You have obtained the account and password with the **Import Data** permissions.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the  icon in the upper-right corner of the page to go to the **Data List** page.

Data List		Model List				
Data Name	Model Name	Upload Time	OLP Type	Mapped OLP	Import Status	Actions
IDCard	IDCard	July 15, 2019 11:12:39 AM CST	Object	IDCard	Success	  

3. Click the  icon next to the specified data to delete the data from Graph Analytics.

8.11. Search

8.11.1. Search

Search is one of the two key modules of Graph Analytics. Research staffs can use the Search module to find and view different objects, such as mobile phones or identity card information. In this topic, you can learn about the features and the entry of the Search interface.

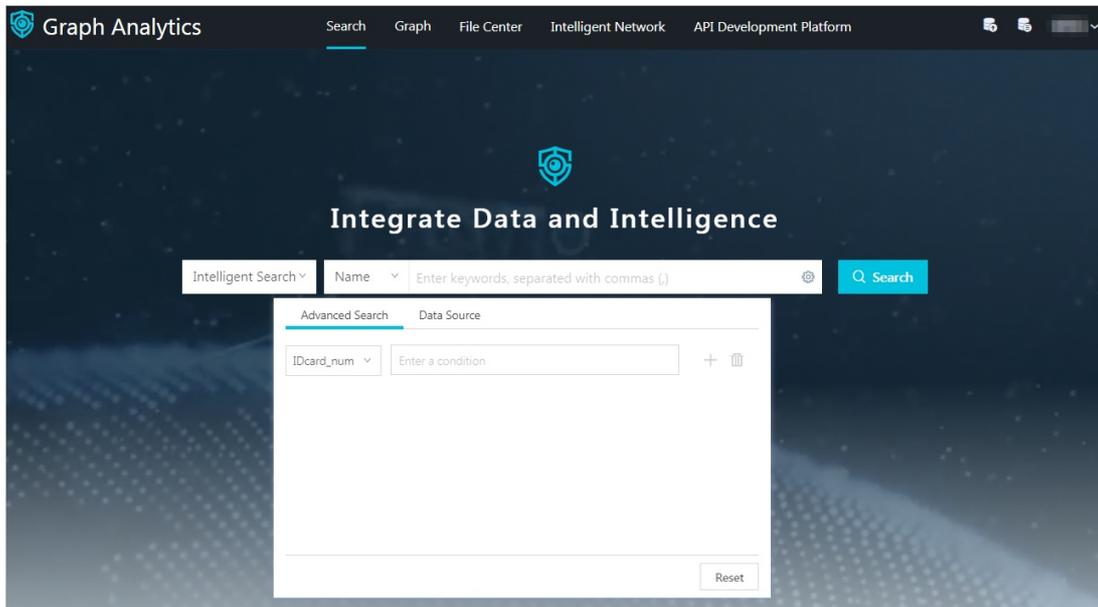
If you have obtained some fuzzy information, you can use the Search module to find objects and link records related to the information. These records can be added as independent object nodes to the relationship analysis, and can be used as a starting point for analysis and decision-making.

In the analysis process, you can expand the information step by step and refine the analysis by keywords, such as name and address. The Search module provides a search tool that allows you to retrieve the object information and locate the target information quickly.

 **Notice** In case of empty search conditions, no results will be returned, and a message Enter at least one condition value. will appear.

The Search module also serves as the entry to the Graph module. You can import the information of retrieved objects to the Graph module for further link extensions and link analyses.

Search



8.11.2. Simple search

You can use this feature to quickly search for objects that contain a certain type of keyword. Fuzzy search is supported.

Prerequisites

A search item has been configured for the target object, and the search item has been associated with the property of the object. For more information, see [Configure a search item](#).

Context

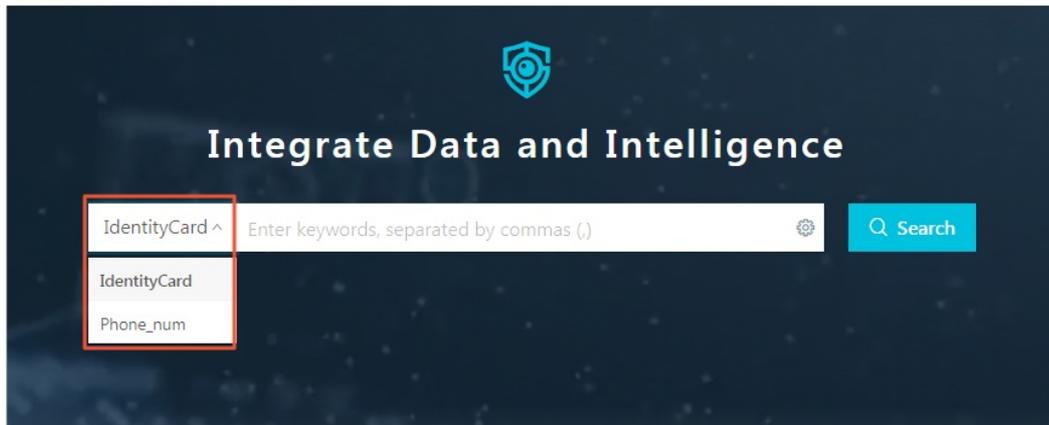
When you perform a simple search, you only need to select a keyword type and enter one or more keywords.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click **Search** on the top navigation bar to go to the **Search** page.
3. Select a search item in the drop-down list as a keyword type, and enter one or more key words based on the search item you select.

Fuzzy search is supported. For example, if you want to search for objects whose phone number contains **138**, you can just enter **138** in the search box.

Search box



4. Click **Search** or press **Enter** to start a search.

phone_num	identity_card	name	Uploaded By	Upload Type	Uploaded At
138xxxxxx1	[REDACTED]	Mr Li	System	System	System
138xxxxxx2	[REDACTED]	Mr Wang	System	System	System
138xxxxxx3	[REDACTED]	Mrs Li	System	System	System
138xxxxxx5	[REDACTED]	Mr Sun	System	System	System
138xxxxxx4	[REDACTED]	Mr Liu	System	System	System

8.11.3. Advanced search

Advanced Search supports fuzzy search and multiple search conditions.

Prerequisites

You have configured a search item with an advanced association item for the target object, and the search item has been associated with the property of the object. For more information, see [Configure a search item](#).

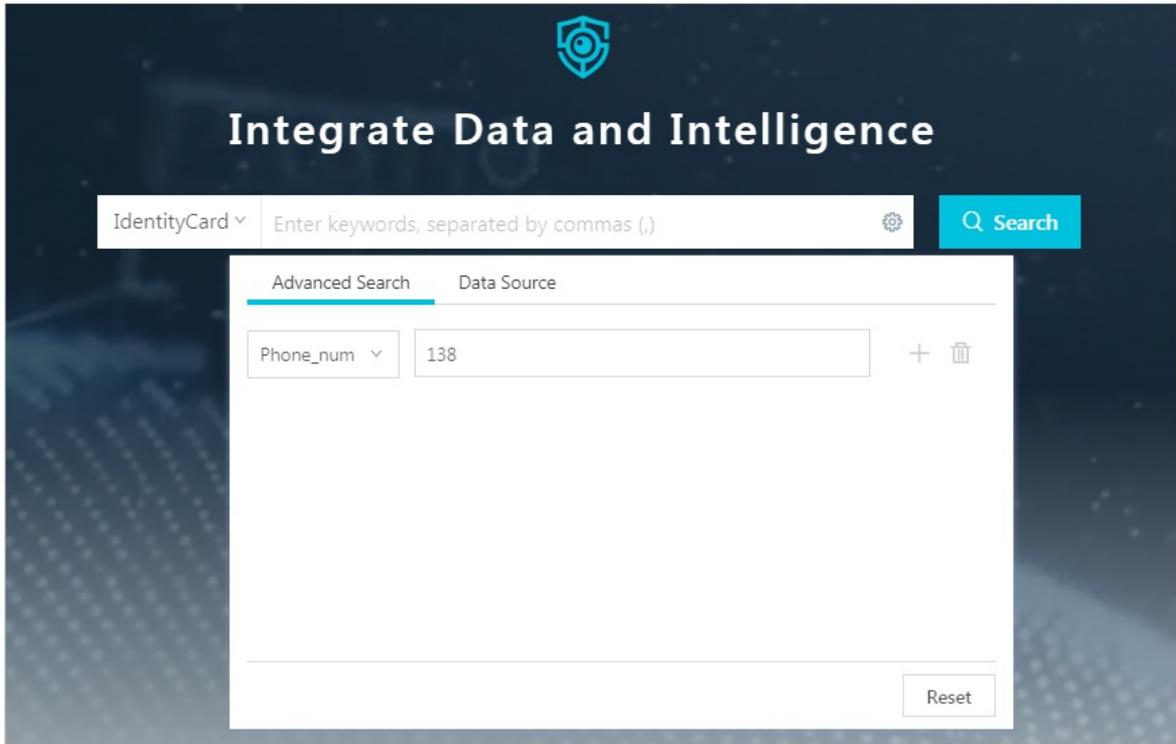
Context

You can specify the search terms in Advanced Search in the same way you perform a simple search. You can specify the advanced correlated items for the selected search terms. This is similar to a combined search based on multiple keyword types. You can also specify the data source items to be searched, which is similar to specifying the search range.

This topic describes a sample search for objects named **liqing344** whose identity card contains **234**.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click **Search** on the top navigation bar to go to the **Search** page.
3. Select a search item in the drop-down list as a keyword type, and enter one or more key words based on the search item you select. In this example, the search item is set to **ID Card** and the keyword is **234**.
4. Click the icon next to the search box to set the condition to perform an advanced search.

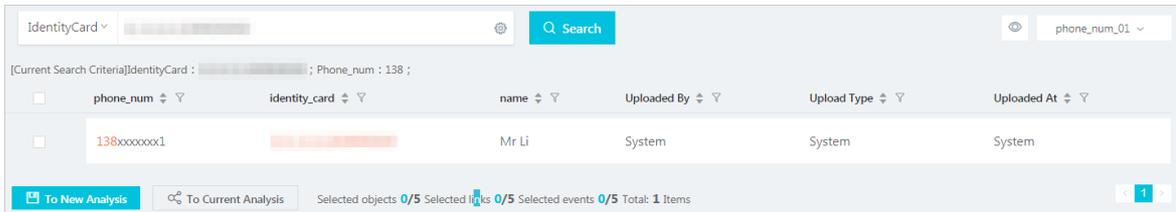


Advanced Search displays the Advanced Correlated Items of the currently selected item. In this case, the **advanced correlated item** is set as **Name**, and the keyword is **liqing344**.

Data Source: The data sources are the objects and links defined in Graph Analytics. Here you only need to select the data sources you want to search.

Reset: You can reset the search condition to the initial status.

5. After you have configured these parameters, click **Search** or press **Enter** to start a search.



8.11.4. View and analyze search results

After you have completed a search, you can view the search results and send the specified content to the graph to perform an analysis.

Prerequisites

You have completed a search, and the search results are not empty. For more information, see [Simple search](#) or [Advanced search](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Search for objects and view the search results. For more information, see [Simple search](#) or [Advanced search](#).

The search results for this example are as follows:

Phone_num 138 Search phone_num_01

[Current Search Criteria]Phone_num : 138 ;

<input type="checkbox"/>	phone_num	identity_card	name	Uploaded By	Upload Type	Uploaded At
<input checked="" type="checkbox"/>	138xxxxxx1	[REDACTED]	Mr Li	System	System	System
<input checked="" type="checkbox"/>	138xxxxxx2	[REDACTED]	Mr Wang	System	System	System
<input checked="" type="checkbox"/>	138xxxxxx3	[REDACTED]	Mrs Li	System	System	System
<input type="checkbox"/>	138xxxxxx5	[REDACTED]	Mr Sun	System	System	System
<input type="checkbox"/>	138xxxxxx4	[REDACTED]	Mr Liu	System	System	System

To New Analysis To Current Analysis Selected objects 3/5 Selected links 0/5 Selected events 0/5 Total: 5 Items

3. Select part of or all of the search results. A total of 10 records are displayed on the current page. Click the  icon in the upper-right corner. The search results only show the information of the selected objects.

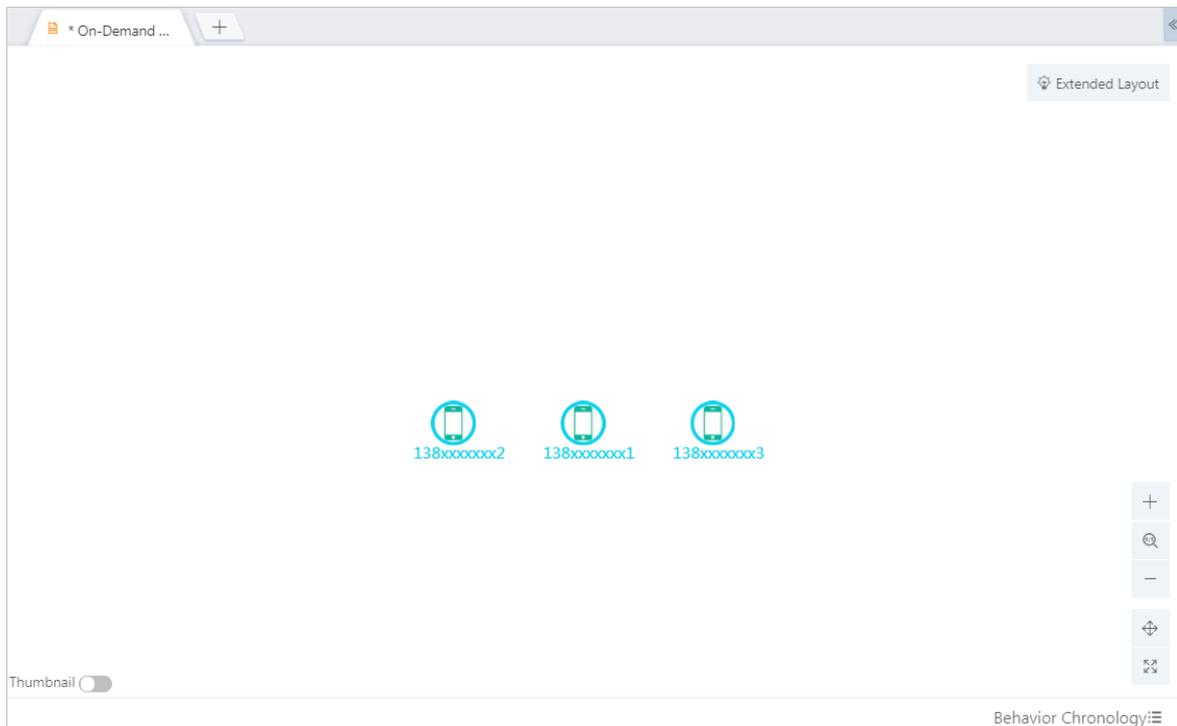
Phone_num 138 Search  phone_num_01

[Current Search Criteria]Phone_num : 138 ;

<input checked="" type="checkbox"/>	phone_num	identity_card	name	Uploaded By	Upload Type	Uploaded At
<input checked="" type="checkbox"/>	138xxxxxx1	[REDACTED]	Mr Li	System	System	System
<input checked="" type="checkbox"/>	138xxxxxx2	[REDACTED]	Mr Wang	System	System	System
<input checked="" type="checkbox"/>	138xxxxxx3	[REDACTED]	Mrs Li	System	System	System

To New Analysis To Current Analysis Selected objects 3/5 Selected links 0/5 Selected events 0/5 Total: 3 Items

4. Click the **To New Analysis** icon in the lower-left corner, or the **To Current Analysis** icon. Send the selected search results to the graph to perform an analysis.



8.12. Graph

8.12.1. Graph

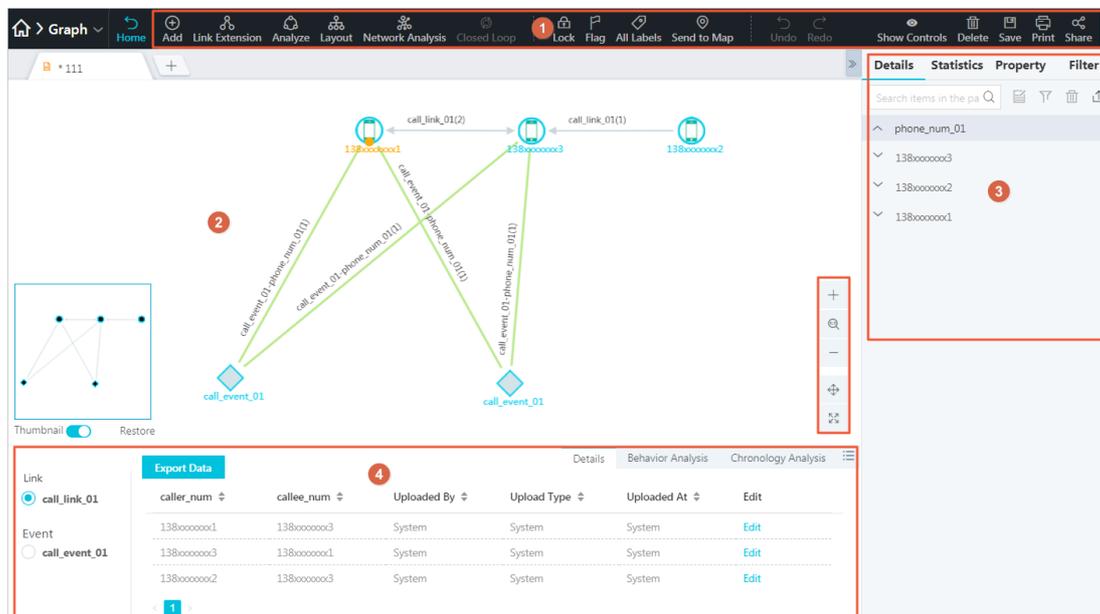
Graph is the operation interface of Analytics Workbench. You can create an analysis and view the analysis results in Graph. In this topic, you can learn about the features, function modules, and other information of Graph.

Graph is the core module of Analytics Workbench. This module covers most of the analysis and decision-making scenarios. The Graph module displays the relationship topologies among all objects. You can perform business computing and interactive graphic operations among multiple objects, and arrange visualized layouts based on your needs.

Meanwhile, user spaces and information cube networks are used as the assisting components to cover a wider range of analysis scenarios.

The Graph interface includes the following four areas.

Graph



- Area 1: functions
- Area 2: main graph area
- Area 3: properties and statistics
- Area 4: behavior analysis and chronology analysis

8.12.2. Analysis types

Graph Analytics supports four types of analyses: temporary analysis, common analysis, shared file analysis, and import data analysis.

Temporary analysis

An analysis that has just been created by the user is known as a temporary analysis. Temporary analyses have the same operations and features as other types of analyses, but temporary analyses cannot be shared before they are saved.

Common analysis

After a new analysis is saved and opened again, it is called a common analysis. The common analysis is the most commonly used analysis in Graph Analytics.

Shared file analysis

The analysis of shared files has the following statuses:

- **Shared file - initial file**

After a common analysis is shared, it becomes a shared file. Each shared file will generate an initial file that is consistent with the original common analysis.

- **Shared file - history analysis (draft)**

After the initial file has been edited by the shared member, a history analysis, also known as draft analysis, will be generated.

- **Shared file - merged file**

The system automatically merges multiple historical analyses into one analysis. Users can also merge the analyses manually.

Import data analysis

The import data analysis imports data in the **Data List** to the graph area to perform analyses. This type of analysis cannot be saved.

8.12.3. Create an analysis

After you log on to Analytics Workbench, you must create an analysis and add objects as nodes before you perform an analysis.

Prerequisites

- You have obtained an account and a password with the permission to perform graphic operations.
- You have created source tables, objects, links, and events.
- You have obtained data in the tables that have been mapped to the primary keys of the objects you want to analyze. You can query the tables in the database to obtain the data.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click **Create Analysis**. A **Temporary Analysis** tab appears.
3. Click **Add** in the toolbar and then click the blank space, or right-click the blank space and select **Add Node**. Specify parameters in the **Add Node** dialog box that appears.

[Parameters to add a node](#) describes the parameters.

Parameters to add a node

Parameter	Description
Object type	<p>Displays all created objects. You can select an object as needed.</p> <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note Graph Analytics allows you to add compound nodes. For example, you can specify two sub-types for object type "person": ID card and passport. The person can be uniquely identified by the ID card or the passport.</p> </div>

Parameter	Description
Text area	Enter one or more primary key values. Separate multiple primary key values with commas (,).

- Click **OK**.
- Right-click a node that has been added, and select **Quick Extension**. The system automatically performs a link analysis based on the configured data sources, objects, links, and events, and displays the analysis results in a graph.

The screenshot displays the Graph Analytics interface. The main area shows a graph with nodes and links. Nodes are represented by phone icons and diamond shapes. Links are labeled with 'call_event_01-phone_num_01(1)' and 'call_link_01(1)'. A table at the bottom shows the 'call_link_01' link data.

Link	caller_num	callee_num	Uploaded By	Upload Type	Uploaded At	Edit
call_link_01	138xxxxxxxx1	138xxxxxxxx3	System	System	System	Edit
call_link_01	138xxxxxxxx3	138xxxxxxxx1	System	System	System	Edit
call_link_01	138xxxxxxxx2	138xxxxxxxx3	System	System	System	Edit

- Select one or more objects, links, or events. Click **Behavior Chronology** in the lower-right corner to view the information on the **Details**, **Behavior Analysis**, and **Chronology Analysis** tabs.
- Select one or more objects, links, or events. Click the icon in the upper-right corner of the right-side pane to view the information on the **Details**, **Statistics**, **Property**, and **Filter** tabs.
- After the analysis is complete, click **Save** in the upper-right corner. In the **Save Analysis** dialog box, specify **File Name**, select a folder, and then click **OK**. A success message appears after the file has been saved.

After you have saved the analysis file, if a collaborative analysis is required, you can share this personal analysis with other members.

- Click the **Share** icon in the upper-right corner to specify the members you want to share this analysis with.

8.12.4. Add a node

Data analyses are based on nodes. Before you perform a data analysis, you need to add the object to be analyzed as a node.

Prerequisites

- Make sure that you have obtained an account and a password with the permission to perform graphic operations.

- You have created a source data table, object definitions, link definitions, and event definitions. For more information about these operations, see [Data sources](#) and [Object information](#).
- Make sure that you have obtained data in the tables that have been mapped to the primary keys of the objects to be analyzed. You can obtain the data by querying the corresponding tables in the database.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis file.
3. Click **Add Node** and click anywhere in the blank space. Or right-click anywhere on the blank space, select **Add Node**, and then set the parameters in the **Add Node** dialog box.

The parameters are described in [Parameter descriptions](#).

Parameter descriptions

Parameter	Description
Object Type	<p>Displays all created objects. You can select an object as needed.</p> <p>Note Graph Analytics supports adding compound nodes. For example, you can specify two sub-types for object type "person": ID card and passport. The person can be uniquely identified by either the ID card or the passport.</p>
Text Area	<p>Enter one or more primary key values in the data table corresponding to the Object Information.</p> <p>Multiple primary key values must be separated by commas (,).</p>

4. After you have configured the parameters, click **OK**.

If you want to add only one node, you can click **Add Node** in the toolbar and click the graph area to place the node. Or, you can right-click anywhere in the blank space and select **Add Node** in the short-cut menu to place the node. When you add multiple nodes, the nodes are placed in a horizontal line layout or a matrix layout by default. You can configure the layout in **Advanced > System Settings > Functional Components > Relationship Network > add multiple node layouts**.

5. Click **Save** in the upper-right corner. In the **Save Analysis** dialog box that appears, enter the **File Name** and select a file directory. Click **OK**. A message is displayed, indicating that you have saved the file.

8.12.5. Delete nodes, links, and events

You can delete the selected content in the current graph area. You can quickly delete the nodes, links, and events.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Context

When you delete object nodes, links and events related to these object nodes are also deleted.

Procedure

1. [Log on to Analytics Workbench](#), and open an existing analysis file, or create a new analysis.
2. You can delete object nodes, links, or events by using the following methods.

Method	Operation
In the toolbar	Select one or more nodes, links, or events in the canvas. Click the Delete icon in the toolbar. A message is displayed, indicating that the item has been deleted.
In the right-click menu	Select one or more object nodes, links, or events in the canvas, right-click a selected item, and select Delete Selected . A message is displayed, indicating that the item has been deleted.

8.12.6. Link extension

You can use link extension to search for all objects that are related to a specific object. You can discover associated clues and intelligence from large amounts of unrelated information, and turn the information into useful intelligence.

Prerequisites

- An analysis file exists. Add a new analysis as shown in [Create analyses](#).
- A node object exists. Add a new node as shown in [Add a node](#).

Context

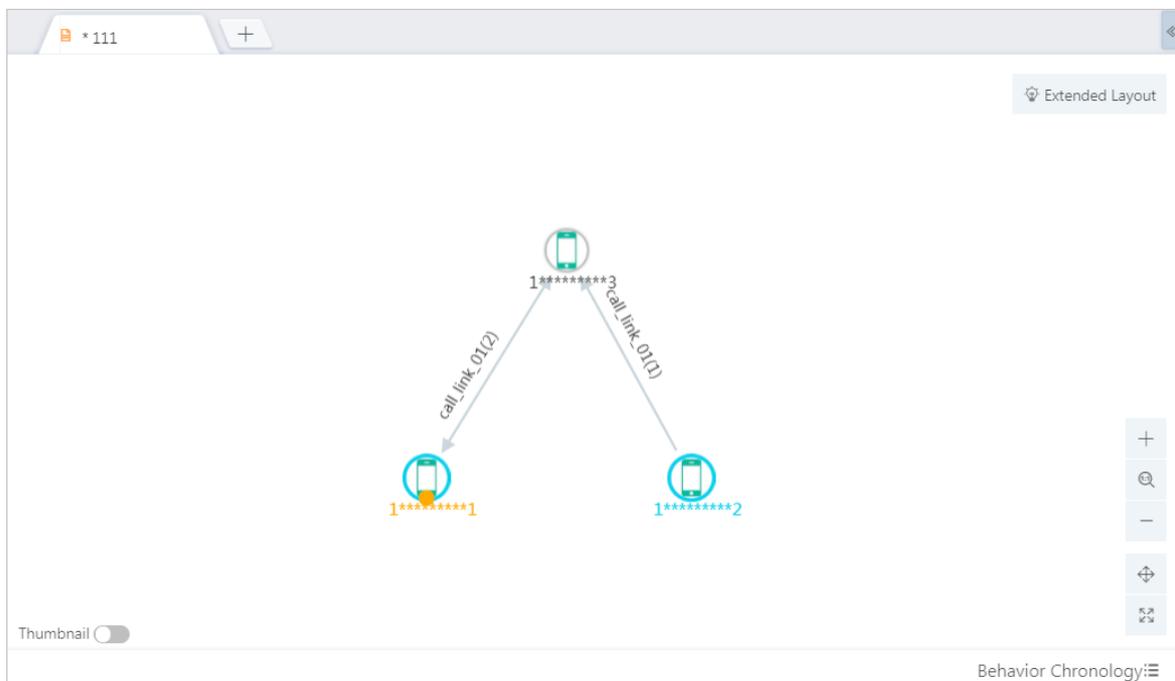
Link extension supports the following two modes.

Link extension mode	Description	Reference procedure
---------------------	-------------	---------------------

Link extension mode	Description	Reference procedure
Simple mode	Double-click a node to perform a first-degree link analysis on the node. Only the associated information for the first-degree link can be extended.	Step 3
Advanced mode	Filter specific conditions based on business experience to extend the associated information.	Step 4 to Step 7

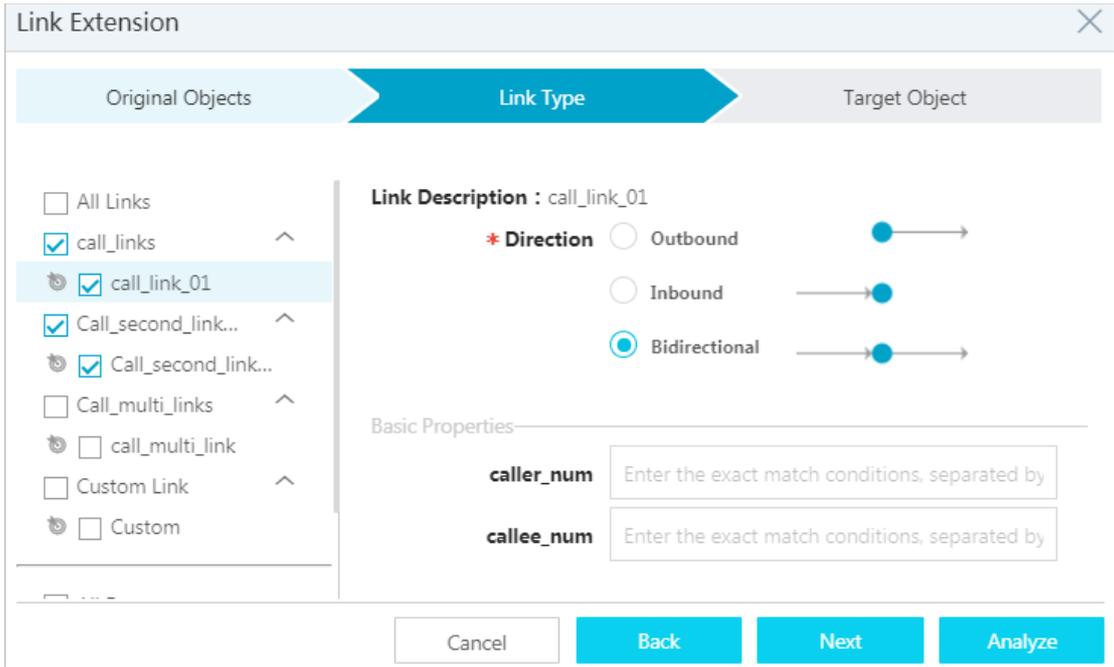
Procedure

1. [Log on to Analytics Workbench.](#)
2. Click an existing analysis file to open the file on the Graph page, or click **Create Analysis** to create an analysis file and add nodes.
Perform a simple analysis on a specific node.
3. Double-click a node to perform a first-degree link analysis.



Perform an advanced analysis on one or more nodes.

4. Select one or more nodes and click the **Link Extension** icon in the toolbar. You can select a maximum of 1,000 nodes. In the **Link Extension** dialog box, select the source objects.
5. Click **Next**, select links and the events, and specify the parameters for the links and events.



6. Click **Next**, and select the target objects.
7. Click **Analyze** to extend the link of the selected nodes.



8.12.7. Graphic operations

8.12.7.1. Move canvases

You can move the entire graph area by moving the canvas to perform comparative analysis.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Procedure

1. [Log on to Analytics Workbench](#), and open an existing analysis file or create a new analysis.
2. You can move the canvas by using the following methods.

 **Note** After you move the canvas, it will take a short time to reproduce the canvas. There will be a short delay. During that time, do not operate the canvas so as to prevent unexpected results.

Method	Operation
In the tool bar	Click the Enter Move Canvas Mode icon  in the lower-right corner of the graph area and drag the canvas with your mouse.
Space key + mouse	Long press the Space key and drag the canvas with your mouse.
Use the scroll wheel	Move the canvas upwards or downwards with the scroll wheel of your mouse.

8.12.7.2. Zoom in and zoom out canvases

Graph Analytics enables you to zoom in and zoom out the content in the graph area to perform a comparative analysis from a full-graph or partial-graph perspective.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Procedure

1. [Log on to Analytics Workbench](#), and open an existing analysis file or create an analysis.
2. You can zoom in or zoom out the canvas in the following methods.

 **Note** After you zoom in or zoom out the canvas, it will take a short time to reproduce the canvas. There will be a short delay. During that time, do not operate the canvas to prevent unexpected results.

Method	Operation
Shift key + mouse pointer	Long press the Shift key and zoom in or zoom out the canvas with the scroll wheel of your mouse.
Interface icons	<ul style="list-style-type: none"> ◦ Click the Zoom In icon  to scale the size of the canvas. ◦ Click the Zoom Out icon  to scale the size of the canvas. ◦ Click the Restore Original Ratio icon  to restore the original ratio of the canvas.

8.12.7.3. Undo and redo operations

Graph Analytics supports undo and redo operations. In case of a mistake, you can quickly restore the graph area to the status prior to the operation.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Background information

The undo and redo functions are described as follows:

- **Undo:** You can temporarily store the graph area status prior to the operation into the operation history. You can restore a historical version by undoing the operation. A maximum of 20 steps can be undone.
- **Redo:** After the Undo operation, you can also restore the analysis to a status prior to the undo operation.

The undo and redo operations can be performed to add nodes, add links, save virtual nodes, merge or split nodes. Undo and redo operations are applicable to layout features, link analysis features, and paste operations, and can be used to delete nodes or links.

Procedure

1. [Log on to Analytics Workbench](#), and open an existing analysis file or create a new analysis to perform multiple operations.
2. If you need to go back to a specific status, you can undo the operation by using the following methods.

Method	Description
Undo	In the toolbar, click the Undo icon to undo the operation and restore the area to the status prior to the operation. You can perform the undo operation at a maximum of 20 steps.
Redo	In the tool bar, click the Redo icon to restore the graph area to the status prior to the undo operation. You can perform the redo operation at a maximum of 20 steps.

8.12.7.4. View thumbnails

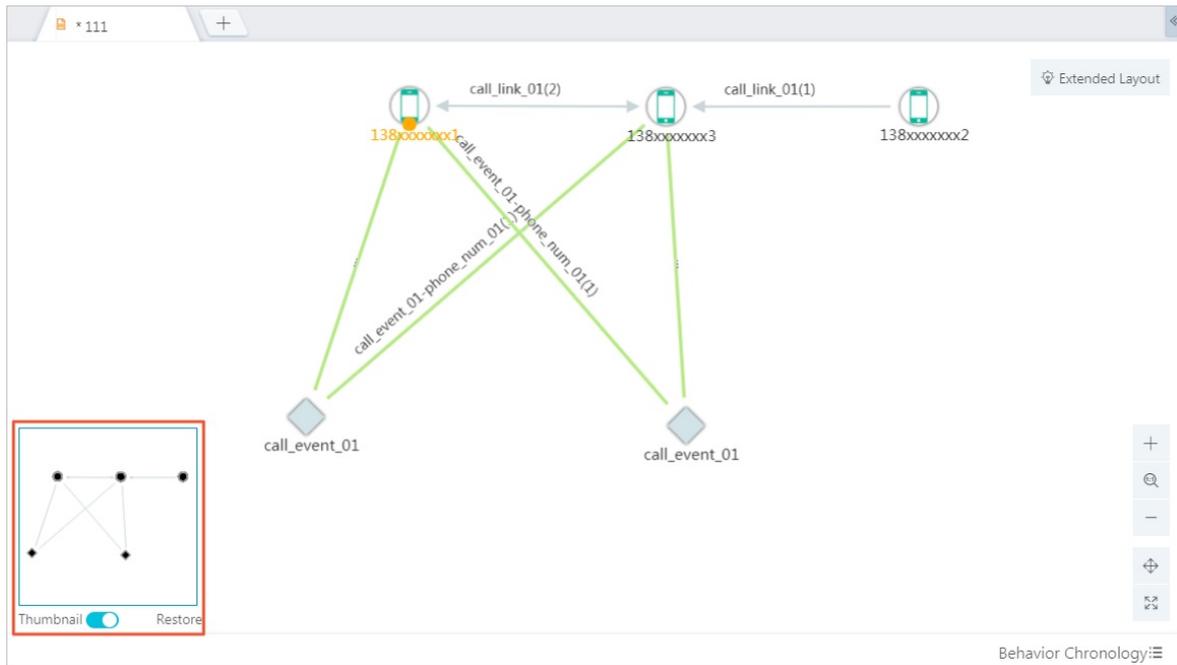
The thumbnail tab is located in the lower-left corner of the graph area. After you click the thumbnail tab, the thumbnail of the current full graph is displayed. The position of the visible area is framed out to help you view the analysis graph easily.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis.
3. Click the **Thumbnail** icon in the lower-left corner to open the thumbnail of the current graph area.



4. You can drag the rectangular box in the middle of the thumbnail to drag the corresponding nodes in the canvas.

8.12.7.5. Right-click menu

The right-click menu in the graph area enables you to quickly operate on nodes, links, or events. In this topic, a vertex refers to a node, and an edge refers to a link or an event.

Log on to [Analytics Workbench](#), and open an existing analysis file or create an analysis. The right-click menu varies depending on the position where you click on.

Right-click anywhere on the blank space

Right-click anywhere on the blank space. For more information, see [Right-click menu description](#).

Right-click menu description

Menu option	Description
Add Node	Adds nodes to the graph area. For more information, see Add a node .
Select All Nodes	Selects all nodes in the graph area.
Select All Events	Selects all events in the graph area.
Select All Links	Selects all links in the graph area.
Paste	Pastes the content in the clipboard to the current analysis. This option will be grayed out if no objects or links exist in the clipboard.

Right-click a node or a link

Right-click a vertex or an edge in the graph area. For more information, see [Right-click menu description](#).

Right-click menu description

Menu option	Description
Add Link	<p>Adds a link for the selected two nodes to establish a relationship network. Graph Analytics supports directed and undirected links.</p> <ul style="list-style-type: none"> • Directed link: A clearly defined link with specific direction. The link line usually has arrows, as in the case of phone call relationship. • Undirected link: A link with no clear direction. For example: people taking a train.
Add Label	<p>Adds label information to the selected nodes for easy analysis and identification. For more information about labels, see Add user labels.</p>
Delete Selected	<p>Deletes the selected nodes or links from the graph area. For more information, see Delete nodes, links, and events.</p>
Inverse Selection	<p>Selects all nodes other than the currently selected nodes.</p>
Select Correlated Nodes	<p>Selects all nodes that are correlated to the currently selected nodes.</p> <p>This option helps users locate the information for analysis and quickly select the correlated nodes. This option applies only to object nodes.</p>
Merge Selected	<p>Merges two or more selected nodes and displays the results.</p> <p>Users can sort, merge, visualize, and simplify the information in the entire graph area. This option applies only to object nodes, and the number of selected nodes must be greater than or equal to 2. Meanwhile, after the Merge Selected option is applied, the object nodes and links in the Selected state in the graph area remain unchanged.</p>
Split Selected	<p>Splits the merged nodes and displays the results.</p> <p>Users can extract information from a merged node for individual analysis. This option applies only to object nodes. Make sure that at least one node in the selected nodes is part of a merged node. Meanwhile, after the Split Selected option is applied, the object nodes and links in the Selected state in the graph area remain unchanged.</p> <p>The split operation filters the label information and property information of merged nodes by type. An UI example is shown in Split Filtered dialog box, and the parameters are described in Filter descriptions.</p> <p>The split options are described in Split options.</p>
Event Chain	<p>Displays all events that occur on a specific object within a specified time range.</p>
Clear Event Chain	<p>Clears all event chains of an object within a specified time range.</p>
Show Event	<p>Shows an event. When an event involves multiple objects, you can use this operation to show the event and view all objects involved in the event.</p>
Hide Event	<p>Hides an event that has been shown.</p>
Quick Extension	<p>Performs a quick extension on the selected nodes, obtains the graphic results of the quick extension, and makes force-directed layouts.</p>

Menu option	Description
Node Redirect	Redirects to a third-party business system by node type according to the URL configured in Administration Console. Node Redirect is a channel that connects Graph Analytics to other products. For example, if the object information can be viewed only by a third-party system, you can click Node Redirect to view the information in the third-party system. Configure the redirect URL in Administration Console depending on the specific situation.
Copy	Copies the selected object nodes or link graphs and pastes them to the clipboard. You can copy object nodes and link graphs.
Roll Up	If a multi-degree link is established between two objects, multiple intermediary objects are displayed between the two nodes. Select these nodes and click Roll Up to hide these intermediary nodes. After the intermediary objects are hidden, a dotted line is displayed between the two objects to represent the multi-degree link. To show these intermediary objects, press Ctrl and click on the dotted line.
Save Virtual Node	Persists virtual nodes.
View Selected Node Details	You can view the details of a selected node in the right-side pane.

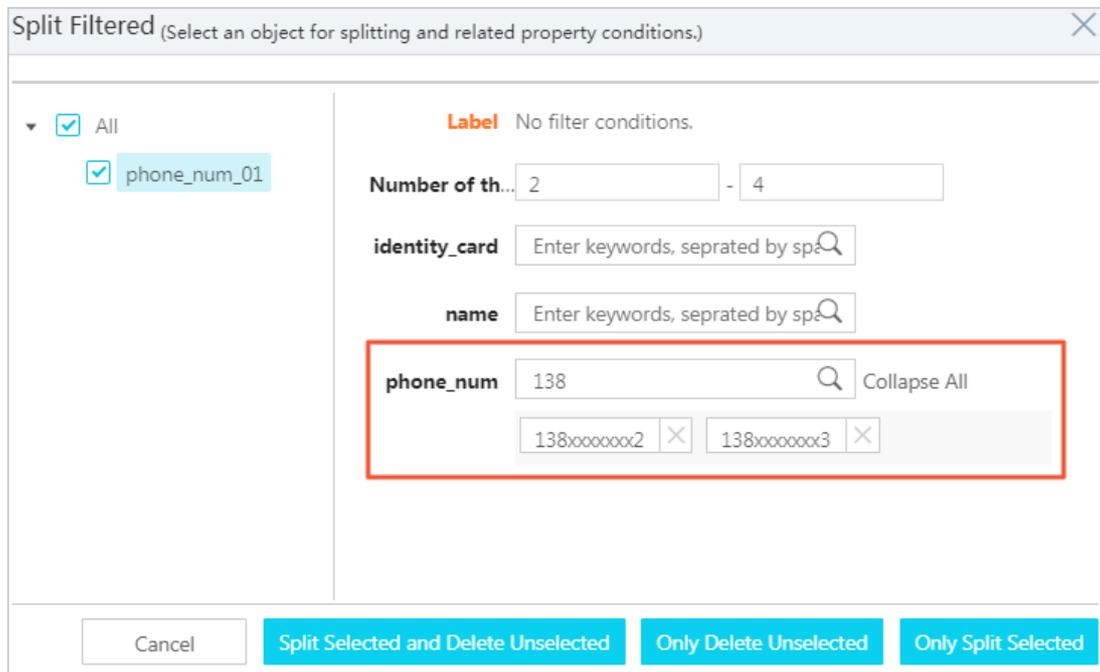
Split Filtered dialog box

Filter descriptions

Filter	Description
Labels	All labels are enumerated. You can delete some labels as needed.
Time type	Lists the maximum and minimum values of the time property. You can adjust the value range, for example, the departure time.

Filter	Description
Value type	Lists the maximum and minimum numerical values. You can adjust the value range, for example, age.
Dictionary type	Like the Label filter, all dictionaries are enumerated. You can delete some dictionaries as needed.
String	Supports fuzzy searches. As shown in Fuzzy search , search for the mobile phone number containing 189. Each item in the search results can be deleted by clicking the Delete icon next to the item. If no results have been found, the message "No results have been found." is displayed.

Fuzzy search



Split options

Option	Description
Cancel	Cancels the split operation.
Split Selected and Delete Unselected	Splits the nodes that meet the conditions, and deletes the other nodes.
Only Delete Unselected	Retains nodes that meet the conditions in the merged node and deletes the other nodes.
Only Split Selected	Splits nodes that meet the conditions from the merged node, and retains the other nodes in the merged node.

8.12.8. Analyze

8.12.8.1. Group Analysis

You can use group analysis to perform multiple analyses on the relationships between any two objects in the group.

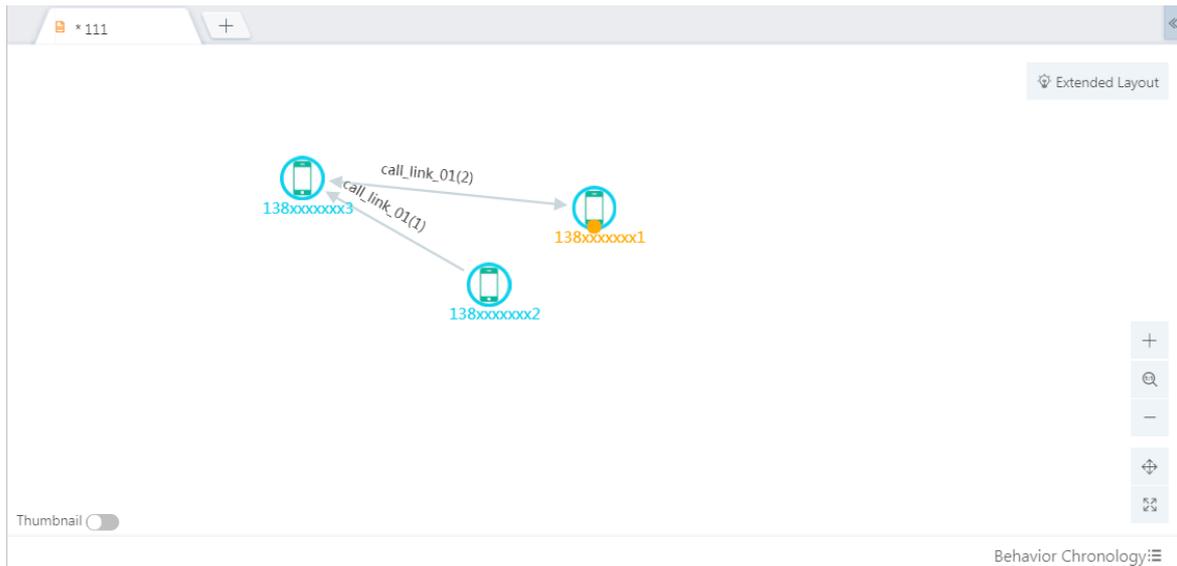
Prerequisites

- Make sure that you have obtained an account and a password with the permission to perform graphic operations.
- An analysis file exists. Add a new analysis as shown in [Create analyses](#).
- Two or more node objects already exist. Add a new object node as shown in [Add a node](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select two or more object nodes in the graph area.
3. Choose **Analyze > Group Analysis**. In the **Group Analysis** dialog box that appears, select the start object to be analyzed.
4. Click **Next**, select links, and set conditions for each link based on your needs.

5. Click **Analyze** to complete a group analysis for the selected node.



8.12.8.2. Common neighbor analysis

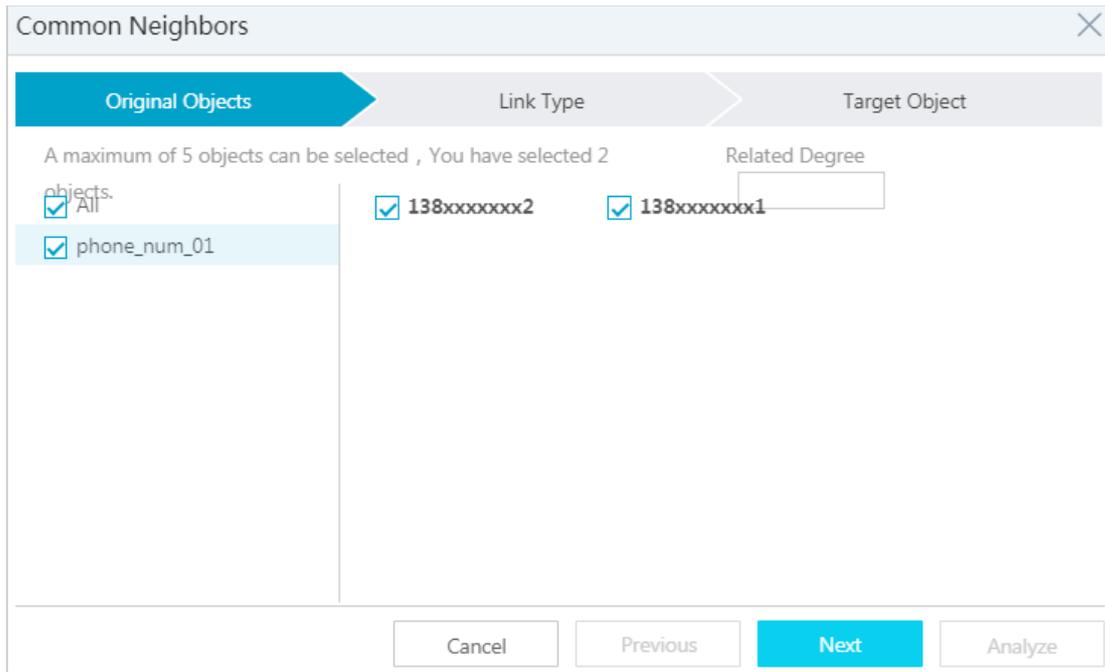
This feature allows you to analyze the commonly associated objects for a group of identical objects or different objects. The correlated degree is set to 2 by default.

Prerequisites

- You have obtained an account and a password with the permission to perform graphic operations.
- An analysis file exists. For more information about how to add an analysis, see [Create analyses](#).
- Two or more node objects already exist. For more information about how to add a node, see [Add a node](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select two object nodes in the graph area.
3. In the toolbar, choose **Analyze > Common Neighbors**. In the **Common Neighbors** dialog box that appears, select the start object to be analyzed.

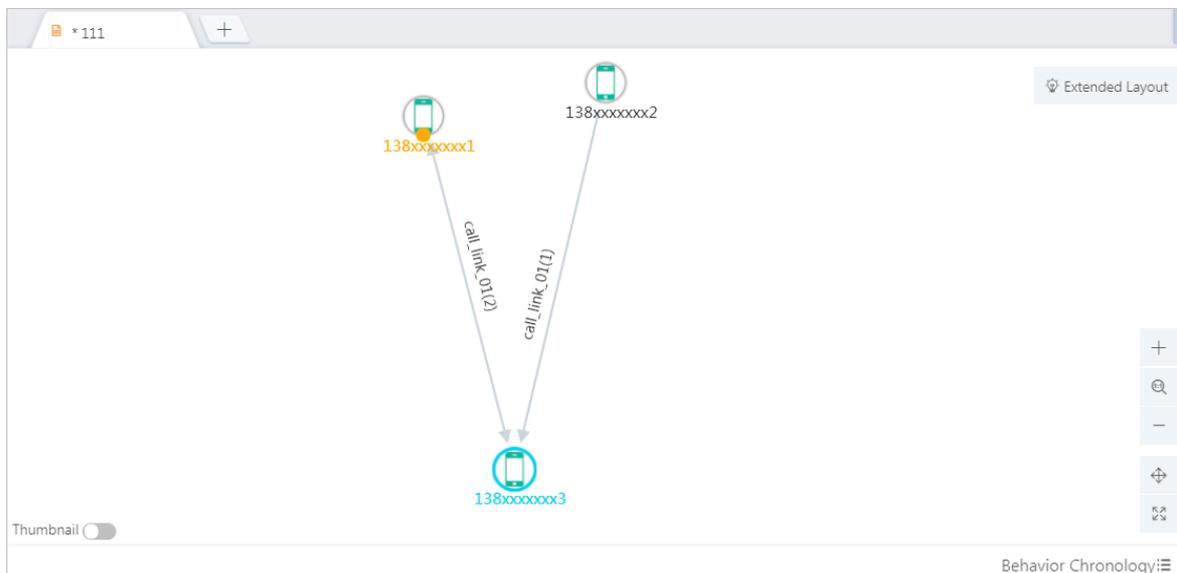


4. Click **Next** to select links and set conditions for each link based on your needs.

The correlated degree is set to 2 by default.

You can set the link conditions by time, date, value, enumerated type, string equality, and fuzzy string matching.

5. Click **Analyze** to complete a common neighbor analysis for the target node.



8.12.8.3. Lineage analysis

Lineage analysis allows you to search the lineage between certain types of objects based on the lineage configurations in Administration Console.

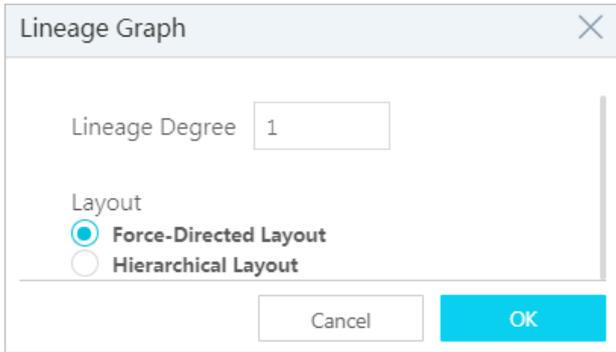
Prerequisites

- You have obtained an account and a password with the permission to perform graphic operations.
- An analysis file exists. For more information about how to add an analysis, see [Create analyses](#).

- A node object exists. For more information about how to add a node, see [Add a node](#).

Procedure

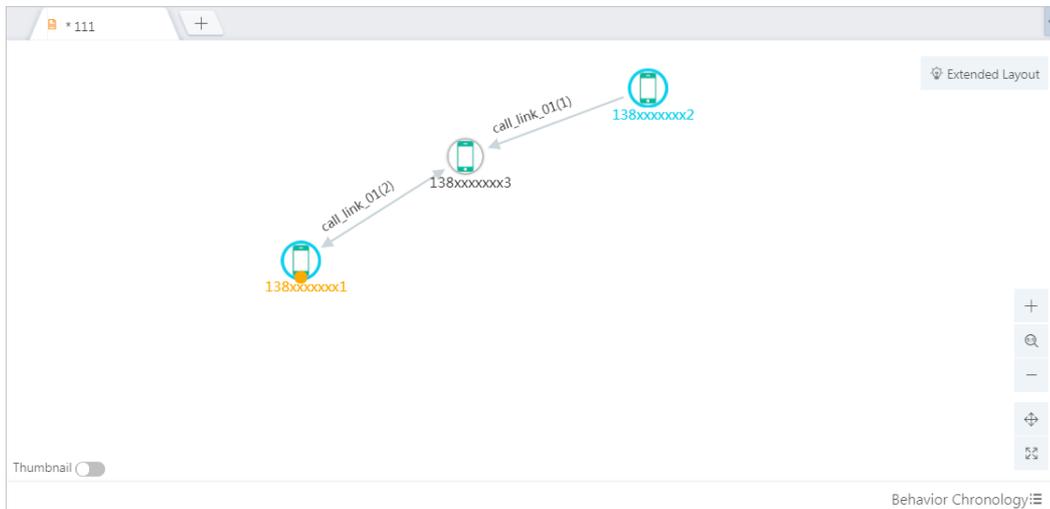
1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select one or more nodes in the graph area to perform a lineage analysis.
3. In the toolbar, choose **Analyze > Lineage Analysis**. In the dialog box that appears, specify Lineage Degree and Layout.



4. Click **OK** to complete the lineage analysis of the target node.

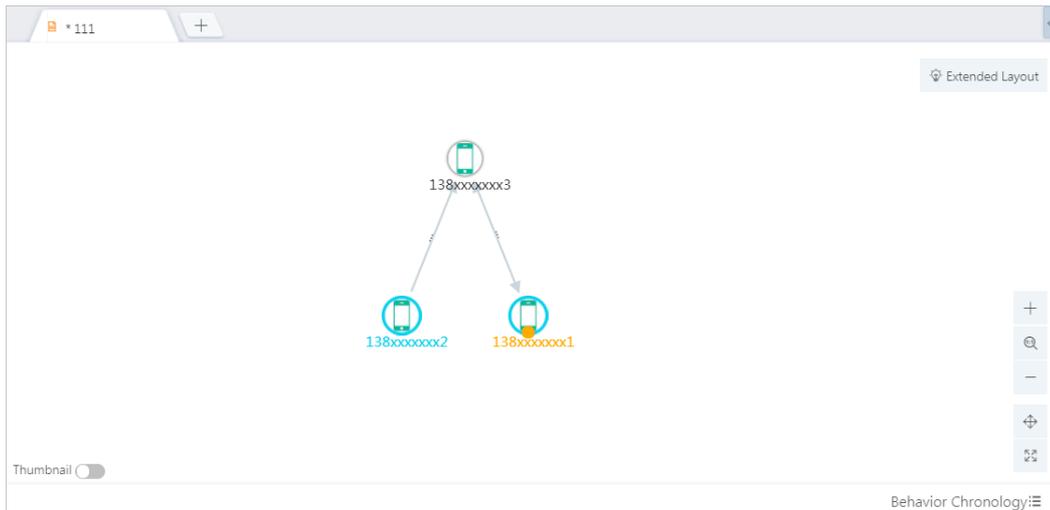
The Lineage Degree is *1*. The Layout is *Force-Directed Layout*. The result is shown in [Analysis results in the force-directed layout](#).

Analysis results in the force-directed layout



Assume that you have set the first-degree link and selected the hierarchical layout. The analysis results are shown in [Analysis results in the hierarchical layout](#).

Analysis results in the hierarchical layout



8.12.8.4. Path analysis

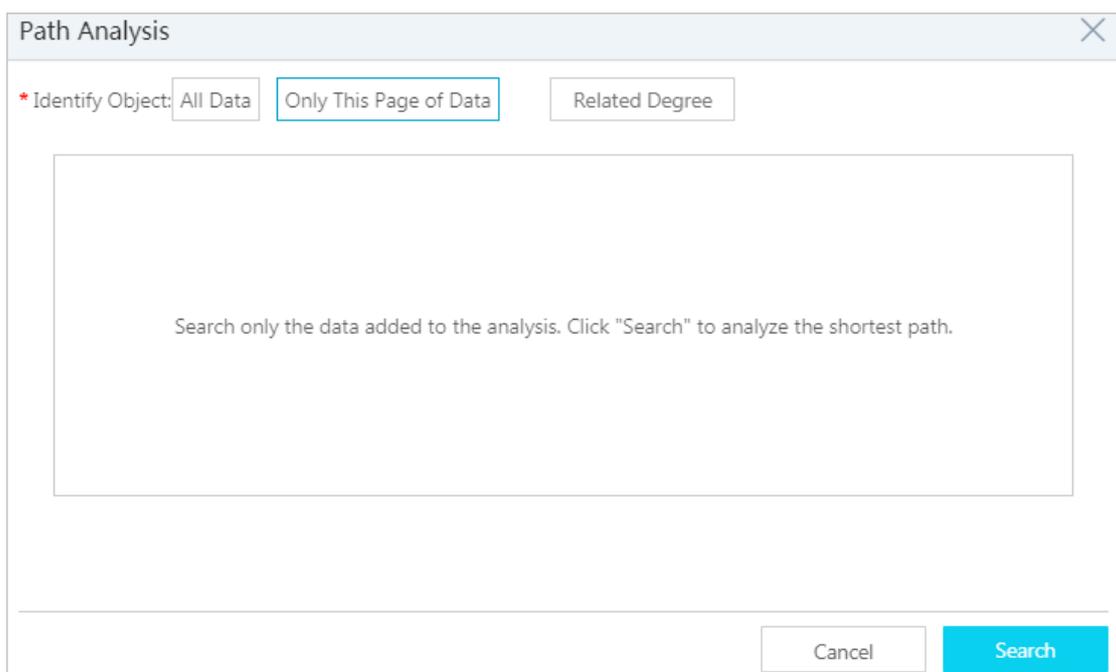
This feature allows you to analyze the link path between two objects.

Prerequisites

- You have obtained an account and a password with the permission to perform graphic operations.
- An analysis file exists. For more information about how to add an analysis, see [Create analyses](#).
- Two or more node objects already exist. For more information about how to add a node, see [Add a node](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select two object nodes in the graph area.
3. In the toolbar, choose **Analyze > Path Analysis**. The **Path Analysis** dialog box appears.



The path analysis supports performing analyses on All Data and Only Data of This Page. For more

information about the path analysis, see the following procedure.

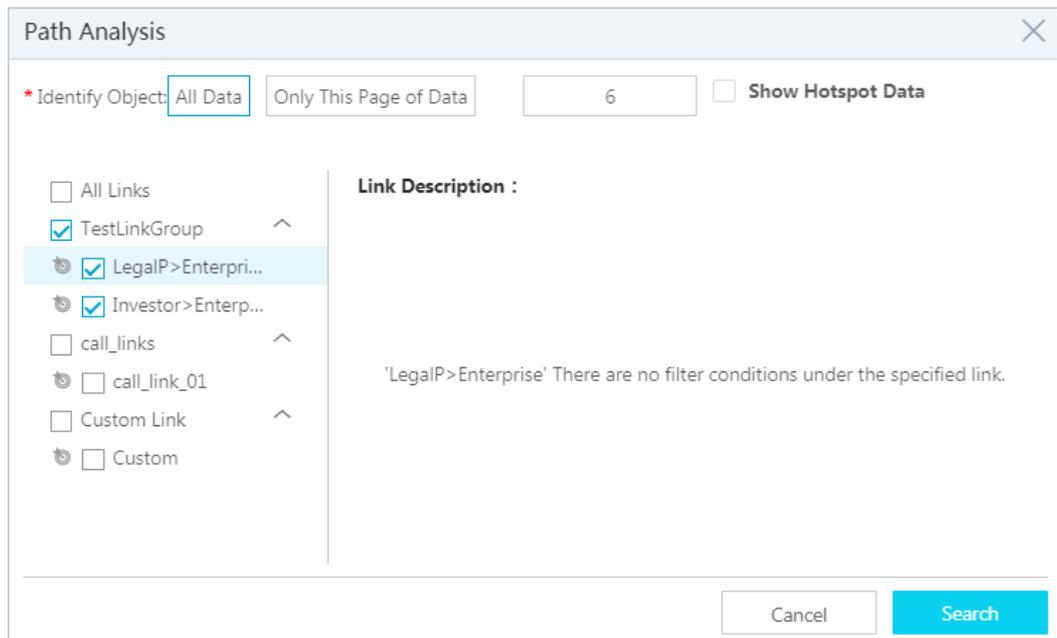
- 4. Perform path analysis on all data: Select **All Data** to perform a path analysis and specify the relevant parameters.

All Data: Set the link condition, correlated degree, and whether to show the hotspot data, as shown in [Set conditions for the path analysis on all data](#).

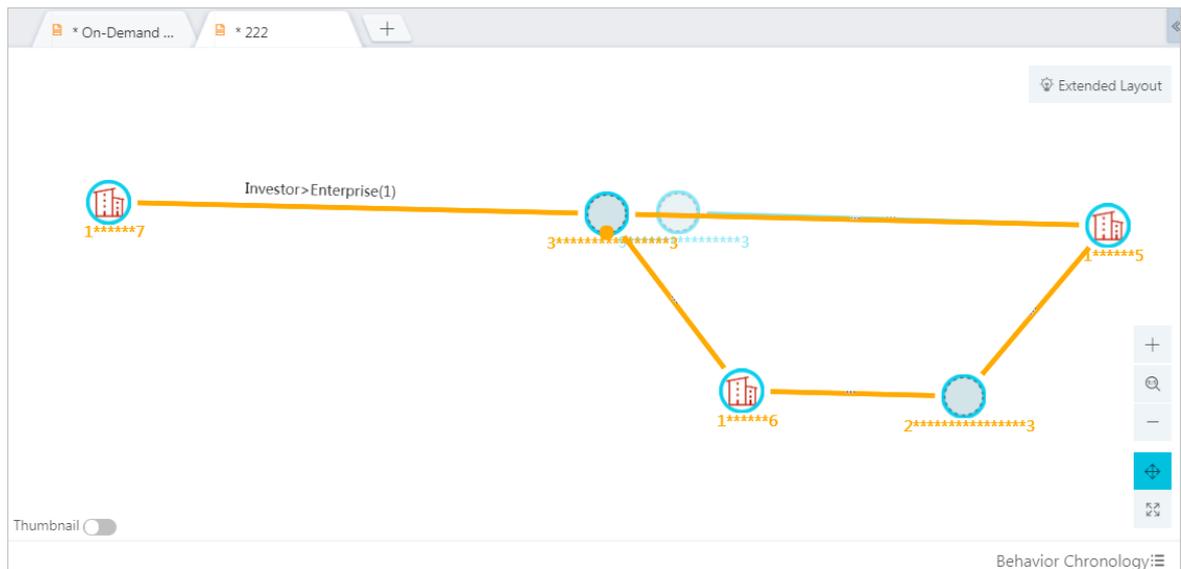
If you select Show Hotspot Data, the hotspot data is displayed but not extended.

Note The system administrator monitors the hotspot data and adds a red tab to the hotspot node to inform the users of the hotspot data.

Set conditions for the path analysis on all data



- 5. Click **Search** to perform a path analysis on all data.



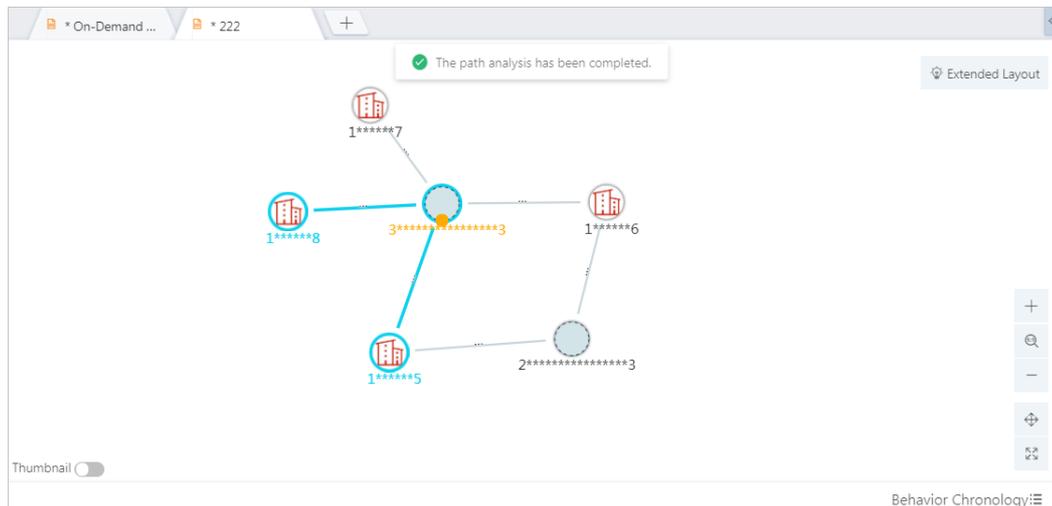
- 6. Analyze only data of this page: Select **Only Data of This Page** and specify the correlated degree.

Analyzing only data of this page is to analyze the path between two nodes on the current page. This feature supports analyzing the shortest path and paths with a specified correlated degree.

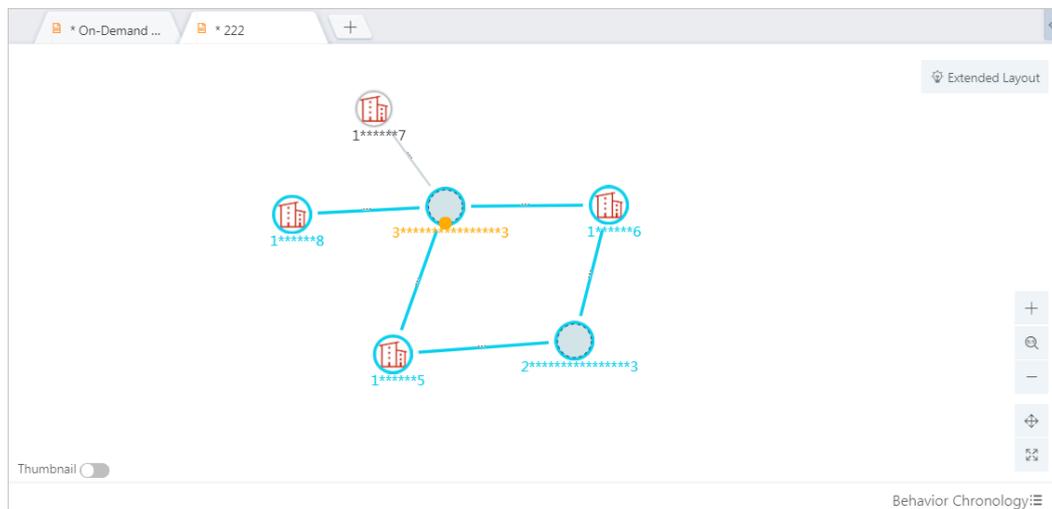
Correlated Degree: If this parameter is not specified, the shortest path is analyzed by default. If you enter N, all paths with degrees less than or equal to N will be searched. N is specified by Administration Console and cannot be higher than 6.

7. Click **Search** to perform a path analysis on the data of this page.

Results of shortest path analysis - only data of this page



Results of the path analysis with a specified correlated degree - only data of this page



8.12.8.5. Backbone analysis

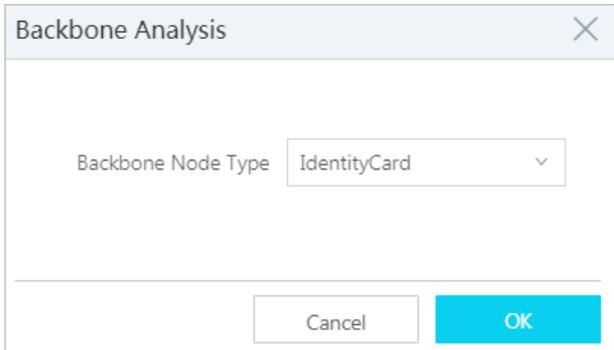
Based on the membership network on the current page, the backbone analysis uses smart service algorithms to help you explore the key nodes in a relationship network.

Prerequisites

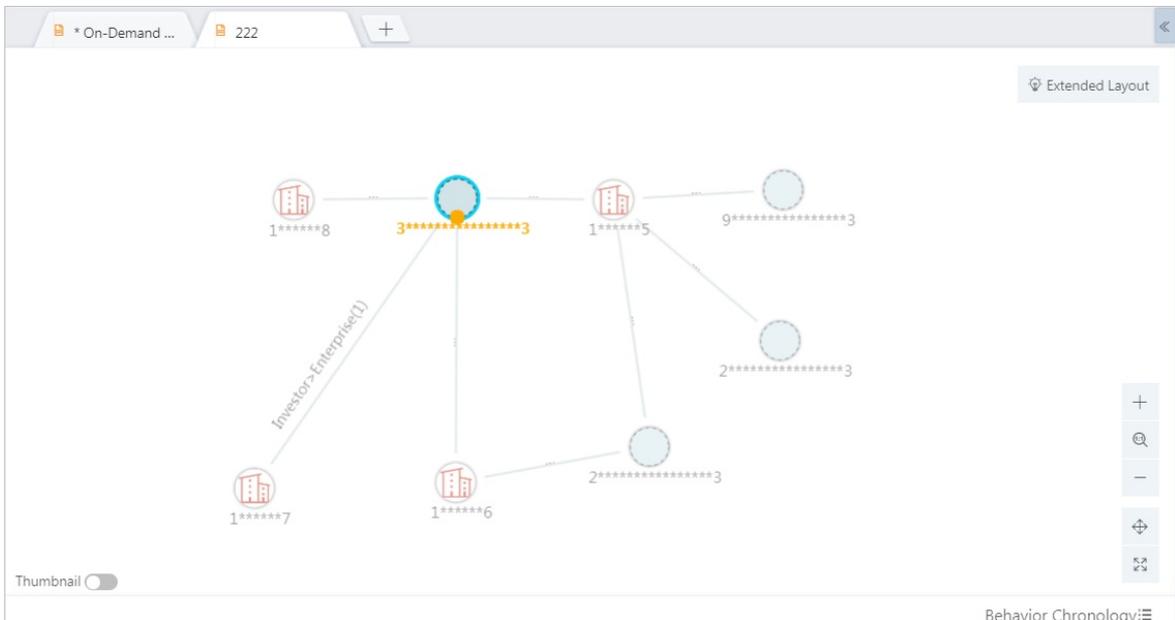
- You have obtained an account and a password with the permission to perform graphic operations.
- An analysis file exists. For more information about how to add an analysis, see [Create analyses](#).
- An object node already exists. For more information about how to add a node, see [Add a node](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select an object node in the graph area.
3. In the toolbar, choose **Analyze > Backbone Analysis**, and set the Backbone Node Type in the dialog box that appears.



4. Click **OK**, and the key nodes in the graph area are highlighted.



8.12.8.6. Intimacy measurements

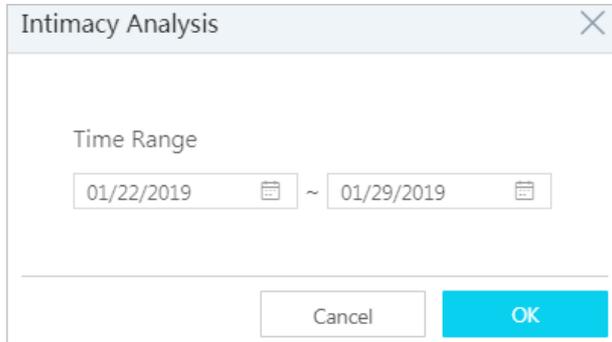
Perform an intimacy measurement to query the intimacy among objects of a specific type based on the intimacy settings configured in Administration Console.

Prerequisites

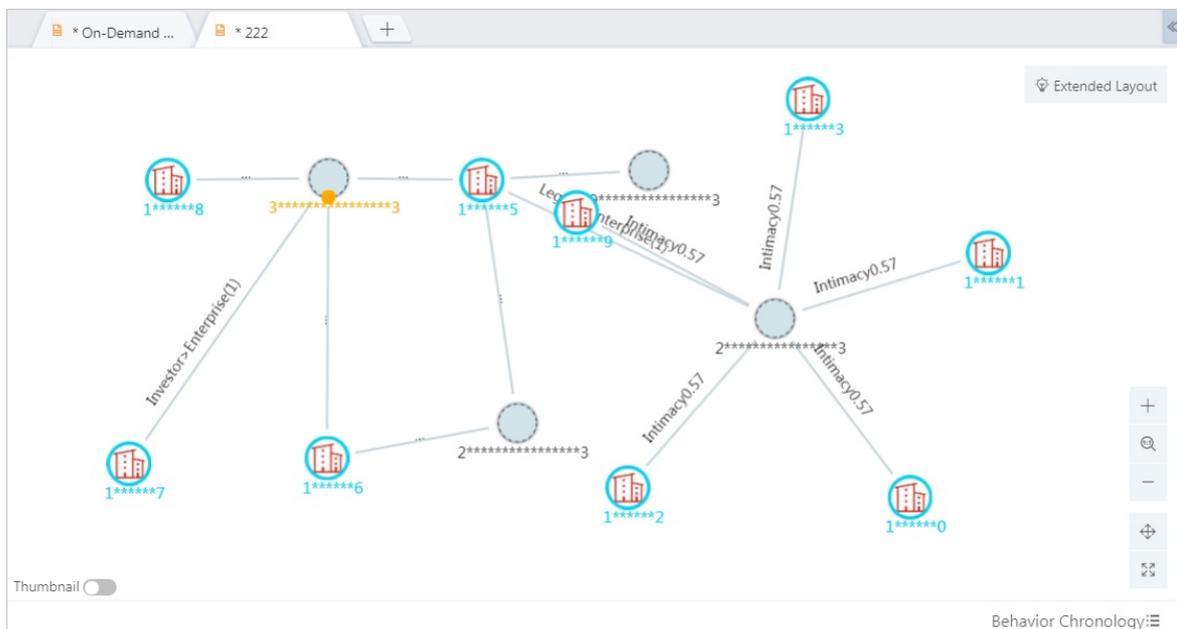
- Make sure that you have obtained an account and a password with the permissions to perform graphic operations and intimacy measurements.
- Make sure that you have configured intimacy settings in Administration Console. For more information, see [Intimacy measurement settings](#).
- An analysis file already exists. For more information about how to create a new analysis, see [Create analyses](#).
- An object node already exists. For more information about how to add a new node, see [Add a node](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create an analysis, and select one or more nodes of the same type that have been created in the graph area.
3. Choose **Analyze > Intimacy Measurement** from the toolbar. In the **Intimacy Measurement** dialog box, set the time range.



4. Click **OK** to perform an intimacy measurement on the selected nodes.



8.12.9. Lock or unlock nodes

The node locking function keeps the node in a fixed position in the canvas to facilitate your operations.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Context

Except for nodes in the **Force-Directed Layout**, nodes in other layouts cannot be dragged. However, when the canvas moves as a whole, the locked nodes will move with the canvas.

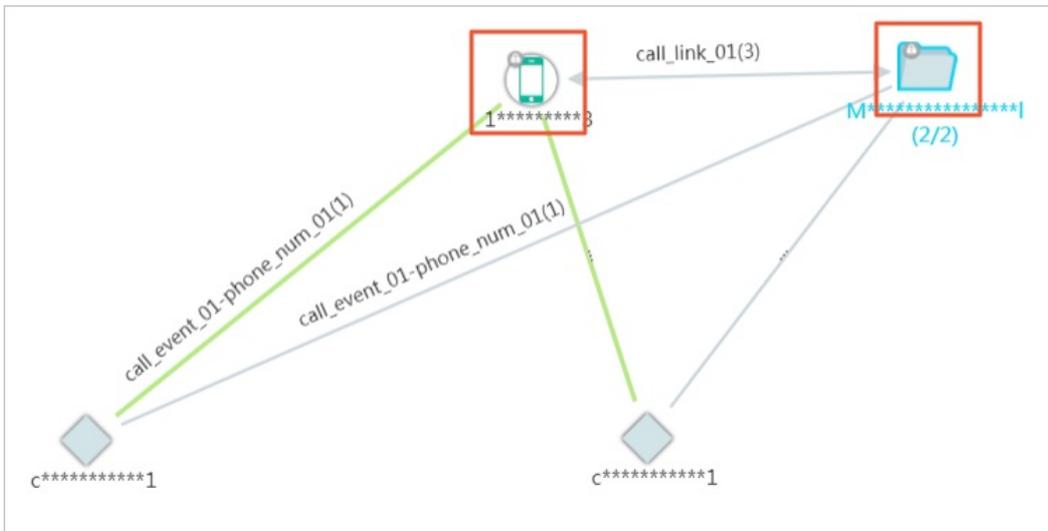
Procedure

1. [Log on to Analytics Workbench](#), and open an existing analysis file or create an analysis.

2. You can lock or unlock nodes by using the following methods.

Method	Operation
Lock nodes	<p>In the canvas, select one or more nodes, including merged nodes, and click the Lock icon  in the toolbar to lock the selected nodes or merged nodes.</p> <p>When a node is locked, you can see a gray lock at the top left of the node, as shown in Lock nodes.</p>
Unlock nodes	<p>In the canvas, select one or more nodes, including merged nodes, and click the Unlock icon  in the toolbar to unlock the selected nodes or merged nodes.</p> <p>When a node is unlocked, the gray lock at the top left of the node disappears.</p> <div style="border: 1px solid #ADD8E6; padding: 5px; margin-top: 10px;"> <p> Note</p> <p>If you split a locked merged node, the split nodes are automatically unlocked.</p> </div>

Lock nodes



8.12.10. Network analysis

Network Analysis can be used to analyze node relationships from multiple perspectives, such as location precedence, closeness, and activity frequency.

Prerequisites

You have obtained an account and a password with the permission to perform network analyses.

Context

The features of network analysis are described as follows:

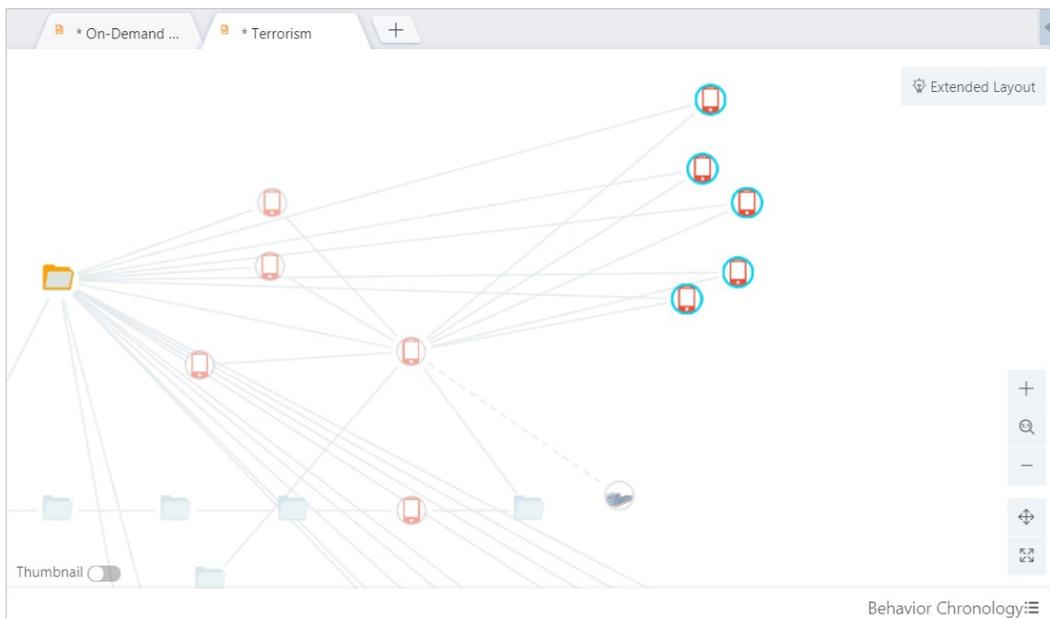
- Closeness: Searches the key nodes that are most closely related to other parts of the network and checks

the statuses of these nodes.

- Location precedence: Searches the bridging nodes that control the information flow.
- Activity frequency: Searches objects that are more active and more frequently interacted with other objects.
- Data flow direction: Searches the flow of relational data between objects in the network. It is only valid for directed relationships.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis, and select one or more object nodes in the graph area.



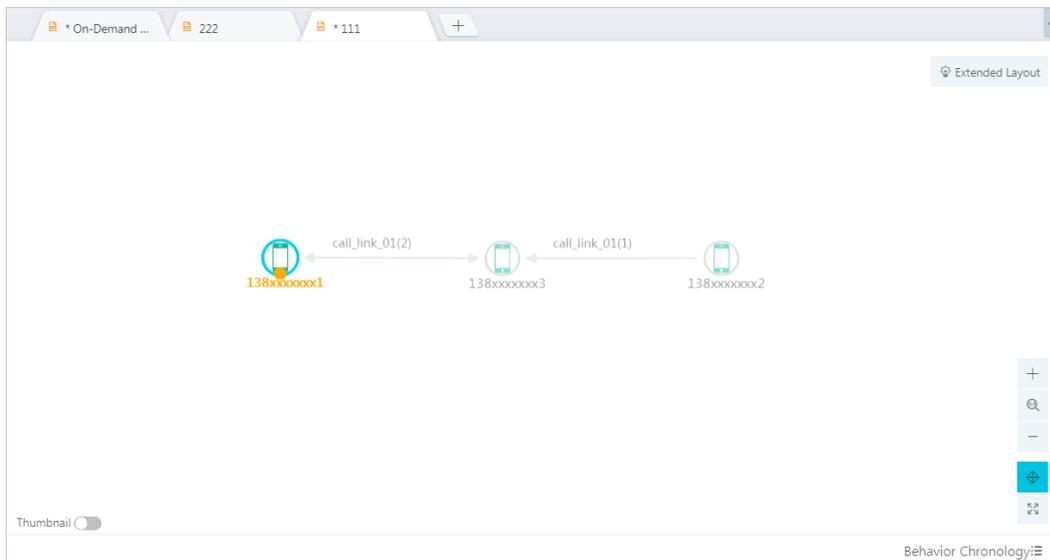
3. Example of location precedence analysis: In the toolbar, choose **Network Analysis > Location Precedence**, as shown in [Location precedence analysis](#).

Location precedence analysis

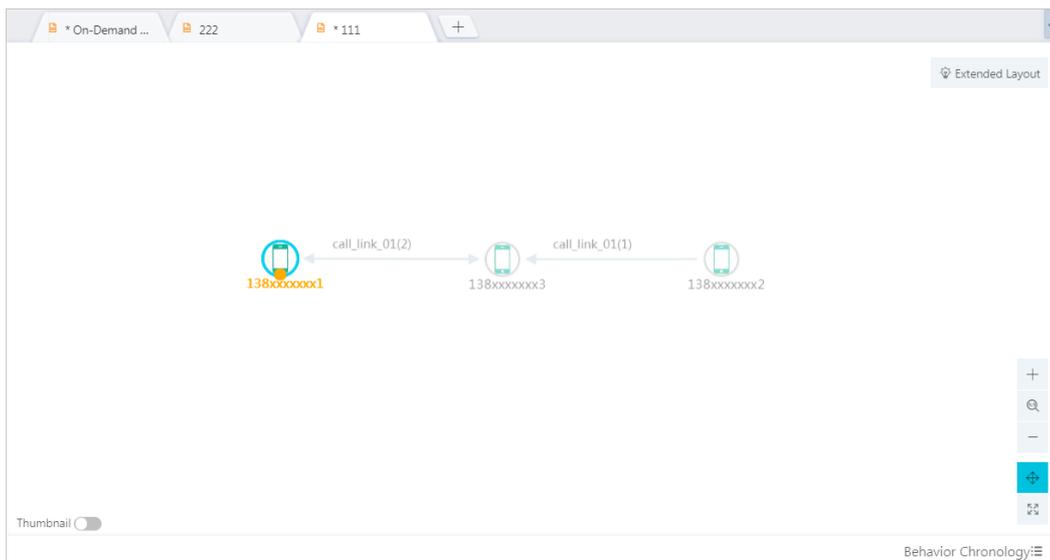


Other analysis examples are shown in [Closeness analysis](#), [Activity frequency analysis](#), and [Data flow direction analysis](#).

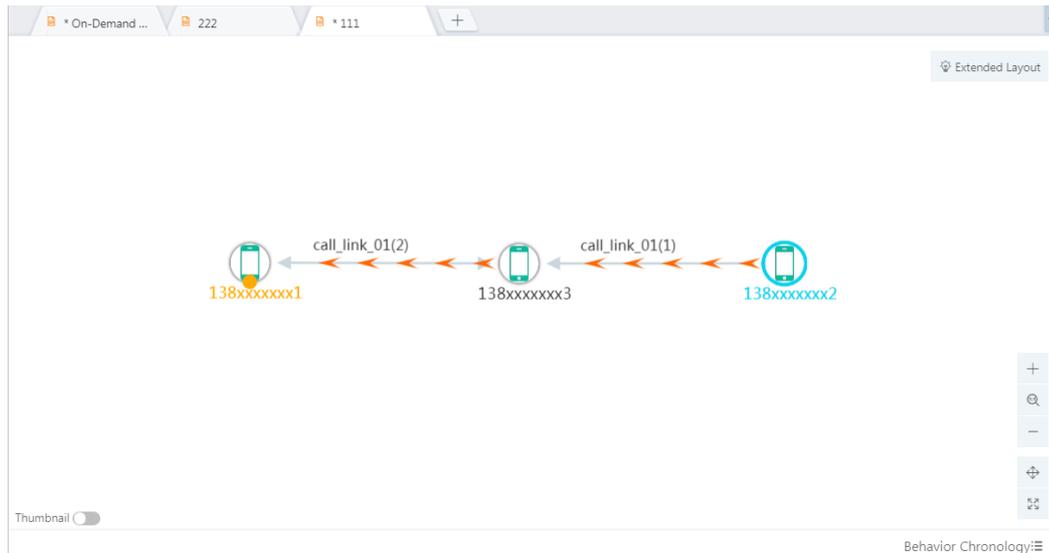
Closeness analysis



Activity frequency analysis



Data flow direction analysis



8.12.11. Closed-loop mining

Closed-loop mining allows you to check one or more nodes for continuous closed-loop links. This feature can be used to analyze cases including cash advance cases.

Prerequisites

- You have obtained the account and password with the permission to perform graphic operations.
- An analysis file already exists. For more information about how to create a analysis, see [Create analyses](#).
- One or more node objects already exist. For more information about how to create a node, see [Add a node](#).

Context

A closed loop refers to a link that starts from a node and ends at the same node. The following closed loops are available: the third-degree closed loops, fourth-degree closed loops, fifth-degree closed loops, and the sixth-degree closed loops.

The following example shows that a credit card cash advance will eventually return to the cashier regardless of the direction of the fund flow. For example, A is the cashier, and both B and C are coordinators. The fund flow is typically $A > B > C > A$, which forms a closed loop.

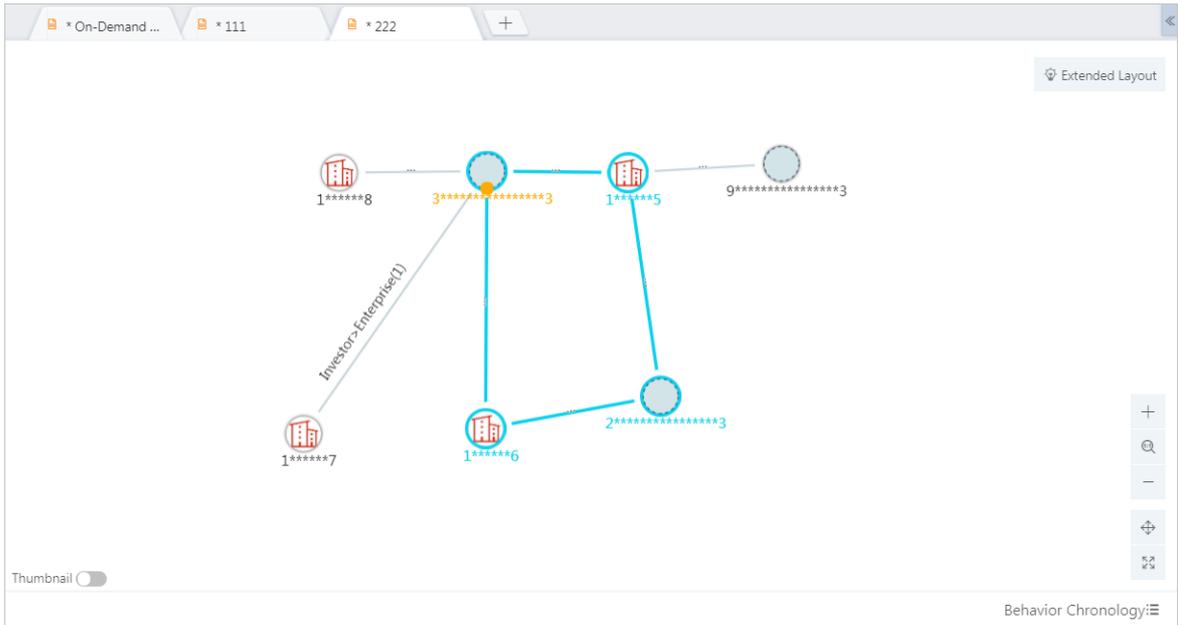
This topic describes loop steps by analyzing the third-degree closed loop of phone calls.

Procedure

1. [Log on to Analytics Workbench](#).
2. Create an analysis and perform related analyses, or open an analysis file that already has analysis results.
3. Select a node in the analysis result, and choose **Closed-Loop > 3-Degree Closed Loop** in the toolbar.
4. In the dialog box that appears, specify **Directionality** and the link, and then click **OK**.

Directionality has the following options:

- *Undirected*: Analyzes both directional and undirectional closed-loop links on the specified node.
- *Directed*: Analyzes only the directional closed-loop links on the specified node.



8.12.12. Layouts

In Graph Analytics, you can easily analyze the content in multiple layouts.

Prerequisites

You have obtained an account and password with **Layout** permissions.

Background

Supported layouts are as follows.

Layout	Description
Matrix Layout	Objects are arranged in a matrix structure to help you sort and organize information during the analysis process.
Circle Layout	Objects are organized and evenly arranged in a circle to display their relationships. This layout helps you sort information and analyze the key nodes during the analysis process.
Horizontal Layout	Objects are arranged along a horizontal line to help you analyze the information from a horizontal perspective.
Vertical Layout	Objects are arranged along a vertical line to help you analyze the information from a vertical perspective.

Layout	Description
Force-Directed Layout	<p>The force-directed layout is used to visualize complex networks. All edges are more or less of equal length and there are as few intersecting edges as possible. With nodes and the weights of edges defined in advance, the force-directed layout positions the nodes automatically according to the principle that a higher weight leads to shorter distance. This procedure is convenient for you to tell how close nodes are from each other.</p> <p>This is a global layout where all object nodes and links in the graph area of Graph Analytics are calculated.</p>
Hierarchical Layout	<p>Objects are arranged in a tree structure. The hierarchical layout is used for family trees and the organizational structures of enterprises.</p>

Procedure

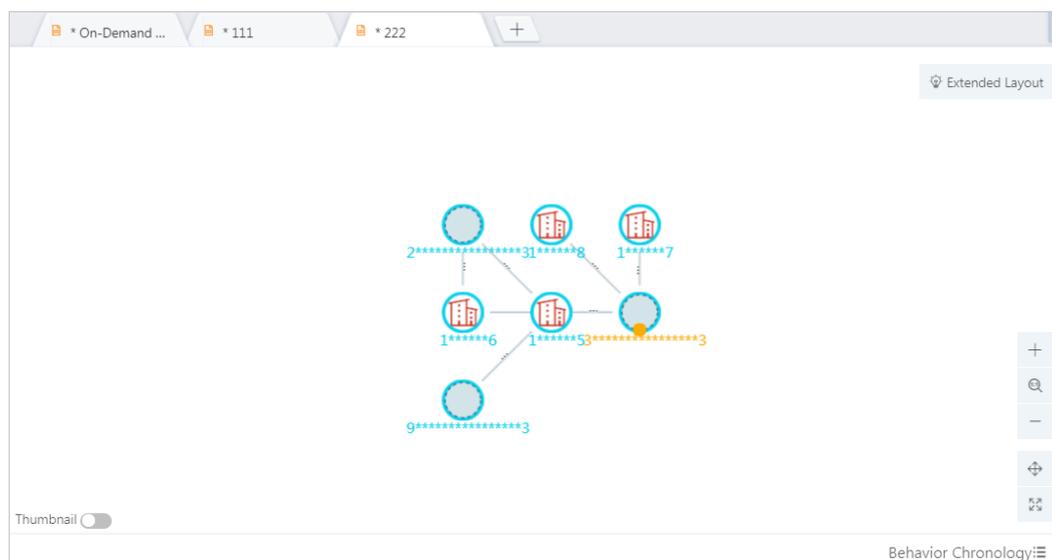
1. [Log on to Analytics Workbench.](#)
2. You can create a new analysis and produce the analysis results. Or, you can open an analysis file that already has the analysis results.
3. Take the matrix layout as an example. In the toolbar, choose **Layout > Matrix Layout**, as shown in [Matrix layout](#).

Note We recommend that you choose a suitable layout for your business, so that you can easily view the analysis data.

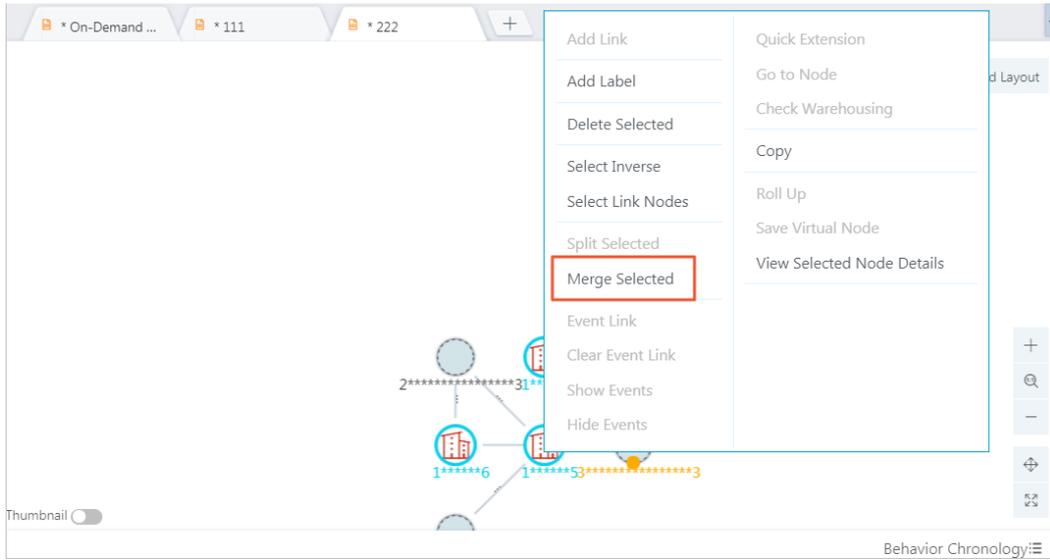
The matrix layout can be used with the Merge Nodes feature, as shown in [Merge nodes](#). This operation can sort and merge the analysis results when you hand large amounts of information. The merged result is shown in [Merged result](#).

Other layouts are shown in [Circle layout](#), [Horizontal layout](#), [Vertical layout](#), [Force-directed layout](#), and [Hierarchical layout](#).

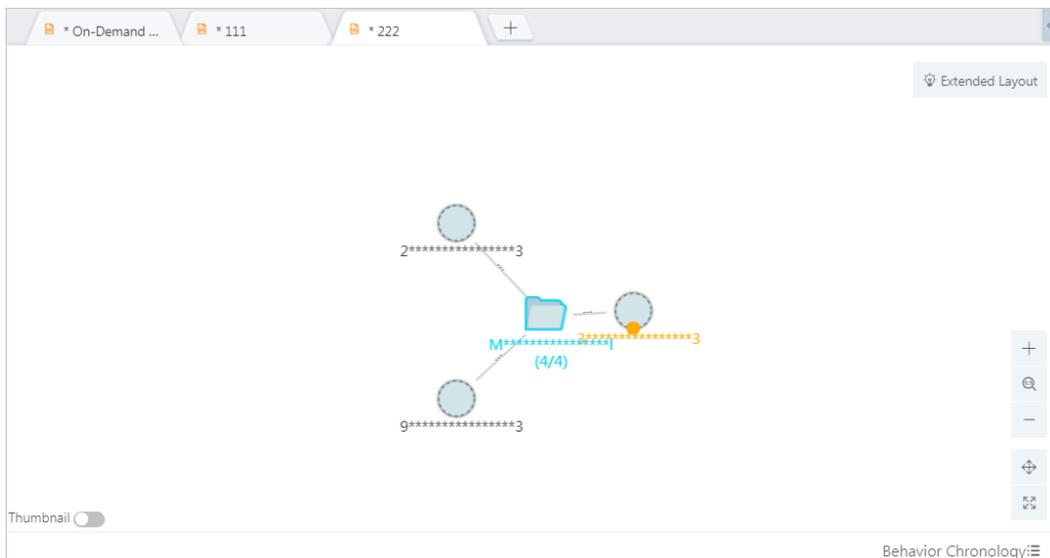
Matrix layout



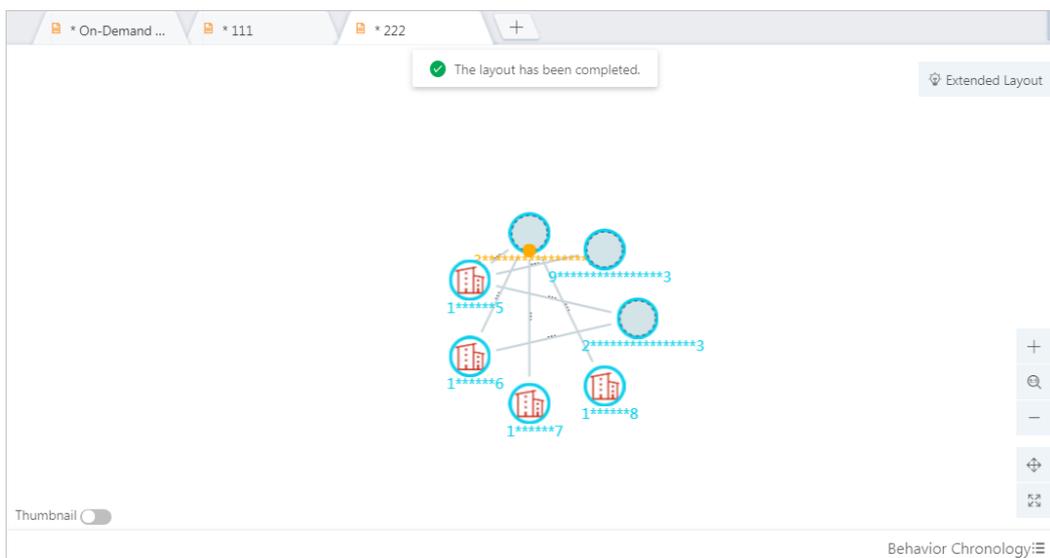
Merge nodes



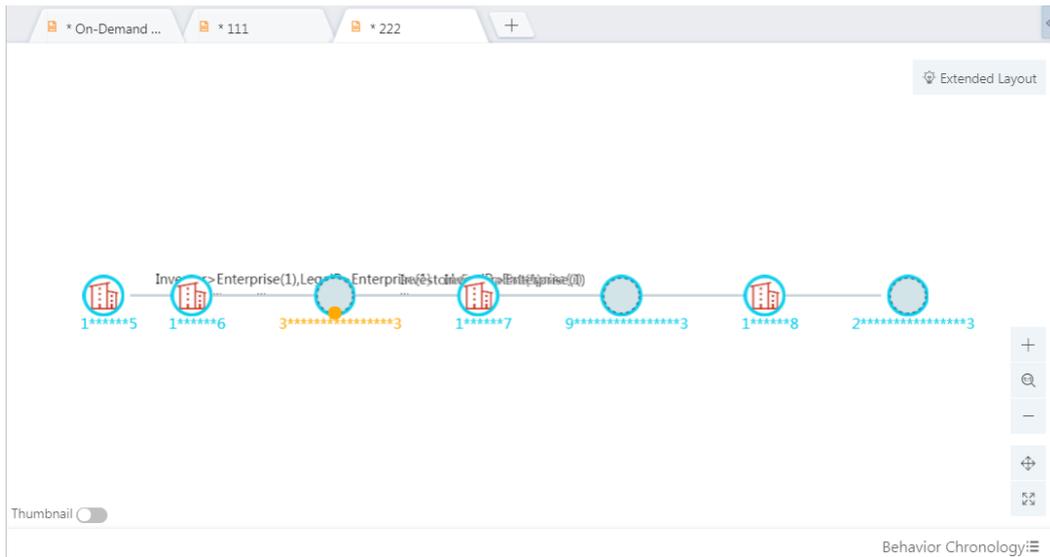
Merged result



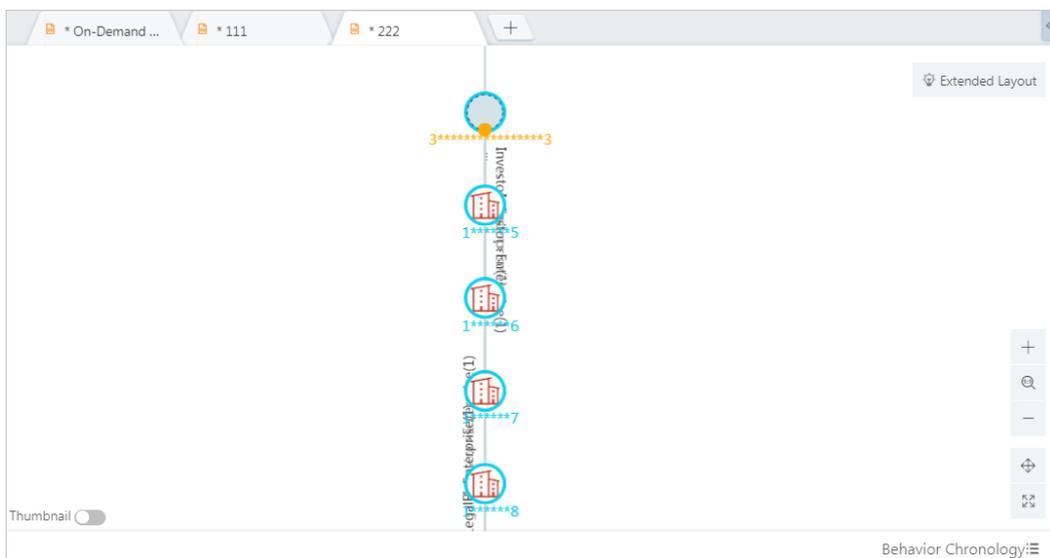
Circle layout



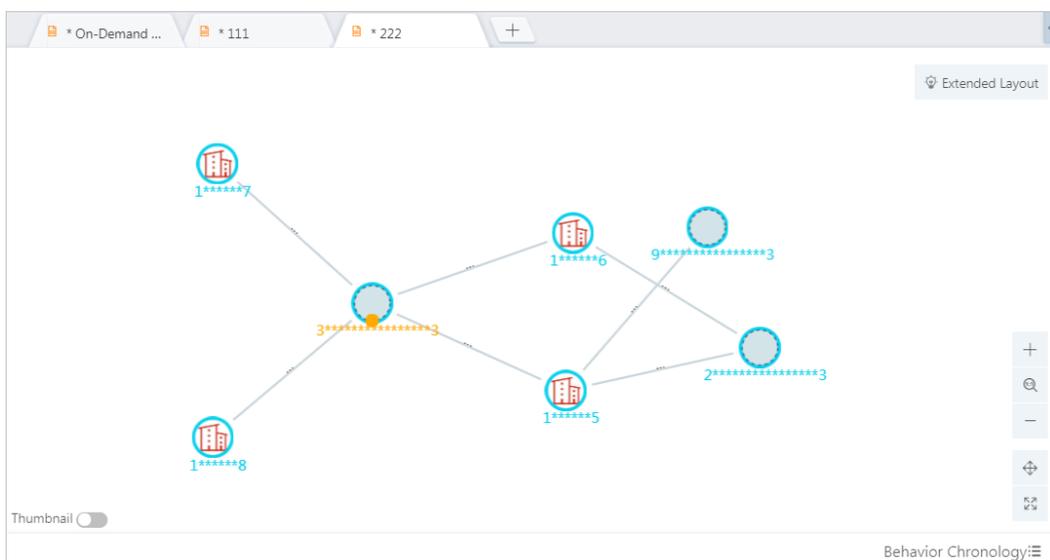
Horizontal layout



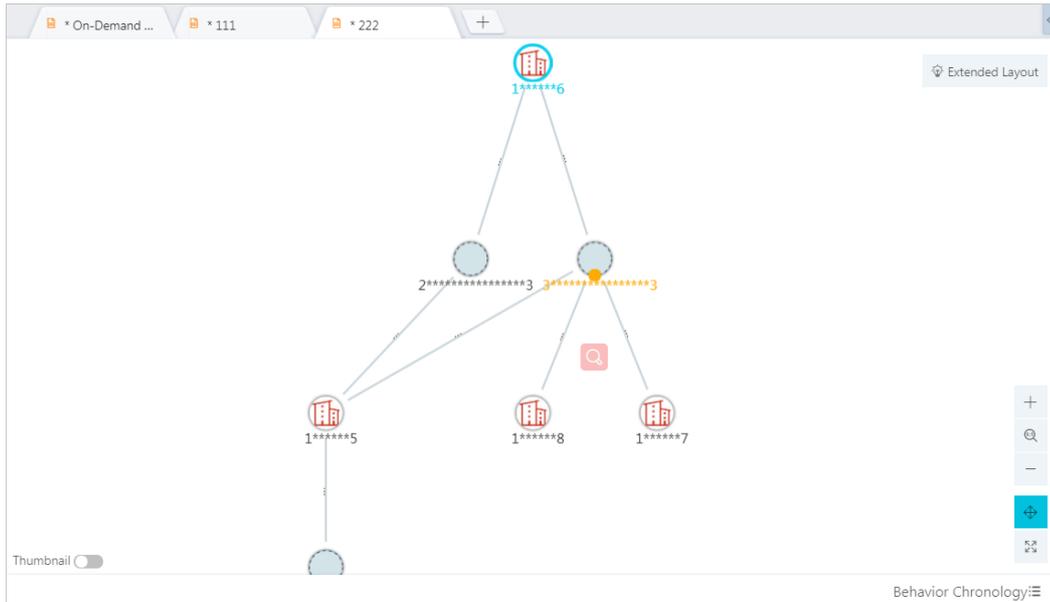
Vertical layout



Force-directed layout



Hierarchical layout



8.12.13. Flag and unflag nodes

When you analyze a large number of object nodes, you can highlight the key object node by adding a small red flag to the node.

Prerequisites

An object node already exists. For more information about how to add a node, see [Add a node](#).

Context

You can perform flag operations on single nodes and merged nodes.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click an existing analysis file to open the file on the Graph page, or click **Create Analysis** to create an analysis file and add nodes.
3. Perform the flag and unflag operations as follows.

Operation	Procedure
Flag	Select the target nodes or merged nodes, and then click Flag in the top navigation bar. A red flag is displayed on the selected node or merged node.
Unflag	Select the target nodes or merged nodes, and then click Unflag in the top navigation bar. The red flag displayed on the selected node or merged nodes disappears.

8.12.14. Labels

8.12.14.1. Label types

You can use labels to identify the content, category, and other properties of the node, which is easy for you and other users to search and locate nodes. In Graph Analytics, you can add label content to the node objects in the graph area. Labels are divided into system labels and user labels.

You can attach a system label or a user label to each node object.

- **System labels:** labels attached to the node objects by the system based on algorithms.

The system labels are displayed on the left side of the nodes. You cannot delete or modify the labels or click the Like button on the system label.

- **User labels:** labels attached to objects by the user.

The user labels are displayed on the right side of the nodes. The user labels with certain permissions can be deleted and liked by corresponding users.

8.12.14.2. User labels

Graph Analytics supports four types of user labels. The visible ranges and operations for labels vary by type.

User label types

- **Public**

After the analysis is shared, all people can see the label and click the Like button on this label.

- **Only Me**

This label is visible only to the person who added the label. After the analysis is shared, other people cannot see this label.

- **Only for This Analysis**

This label is visible only in the current analysis, and the Like button of the label can only be clicked in the current analysis. After a node with this label has been copied to another analysis, this label becomes invisible.

- **Viewable to Specified Members**

This label can only be seen and liked by the users that are specified when the analysis is shared.

User label colors

You can add user labels of different colors. By default, four colors are provided to differentiate the labels: red, yellow, blue, and green, as shown in [User label colors](#).

User label colors

8.12.14.3. Add user labels

When some nodes are hard to understand, you can classify and describe these nodes by adding labels. This helps you and other users to understand these nodes quickly and easily.

Prerequisites

Make sure that you have obtained an account and a password with the permission to perform graphic operations.

Context

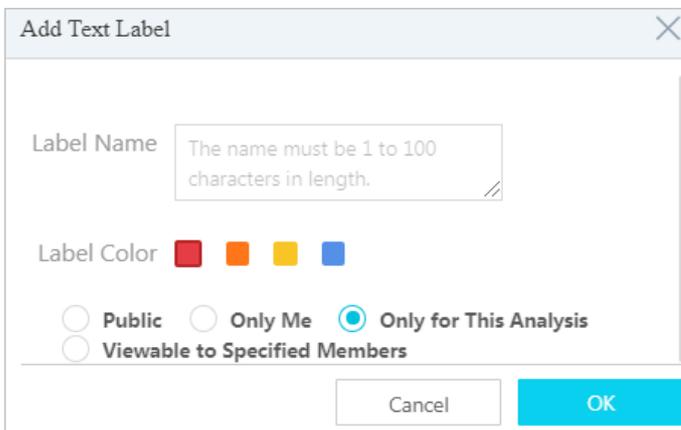
Based on the visible range, labels can be divided into four types: Public, Only Me, Only for This Analysis, and Viewable to Specified Members.

 **Note** You cannot add labels to merged nodes.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Open an existing analysis file or create an analysis.
3. You can use one of the following methods to add a label:
 - o Select one or more nodes in the graph area, right-click a selected node, and then select **Add Label**. Set the parameters in the dialog box that appears.
 - o Select one or more nodes on the **Details** tab of the right-side pane, click the **Text Note** icon in the top of the pane, and then set the parameters in the dialog box that appears.

The parameters are described in [Parameters and descriptions](#).



Parameters and descriptions

Parameter	Description
Label Name	The note or description of the node, which helps you and other users understand the node. The label name must be from 1 to 20 characters in length.
Label Color	The color of the label displayed in the graph area for easy identification. Four colors are supported: red, orange, yellow, and blue.

Parameter	Description
Visible range options	<ul style="list-style-type: none"> ◦ Public: Users who use the node can see the label and click the Like button of the label. ◦ Only Me: The label is visible only to the person who added the label. Other users cannot see the label when they add this node to the graph area. ◦ Only for This Analysis: You can see the label and click the Like button of the label only in the current analysis. This label is invisible after this node is added to or appears in another analysis. ◦ Viewable to Specified Members: Only the specified users can see the label and click the Like button of the label. <p>When you select Viewable to Specified Members, you must specify users as needed.</p>

4. Click **OK**. A success message is displayed after the label has been added.

8.12.14.4. View labels

In Graph Analytics, you can view labels by type and easily analyze the object nodes.

Prerequisites

There are labels visible to you in the current analysis file.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create a new analysis. You can switch the display mode of the labels based on your needs.

Display mode	Operation
All Labels	In the toolbar, click All Labels . The system labels of each node and the labels visible to you in the current analysis are displayed in the graph area.
My Labels	In the toolbar, click the label icon and select My Labels . All labels created by you for the displayed node are shown in the graph area.
Hide Labels	In the toolbar, click the label icon and select Hide Labels . All labels in the graph area are hidden.

8.12.14.5. Click likes and delete likes

When you agree on a label that was added by another user to a node, you can click the Like button for the label.

Prerequisites

- Make sure that you have obtained an account and a password with the permission to perform label-related operations.
- Make sure that other users have added labels that are visible to you.

Context

Only labels that are Public, Only Me, and Viewable to Specified Members can be liked. By default, every label has one like. When the last like of a label is canceled, the label is removed from the node.

After you like a label, the number of likes increases by 1, and the Like button  changes to the Undo Like button . After you undo a like, the number of likes decreases by 1.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Open an existing analysis file or create an analysis, and add or locate a node that has user labels that are visible to you.
3. Click the label number next to the node, and click a label name. The detailed information about the specified label is displayed.



4. If you agree on the label content, click the Like button . A success message is displayed after the operation is completed.

What's next

If you think that a label you previously liked does not match the node, you can click the Undo Like button  to undo the like.

8.12.14.6. Edit user labels

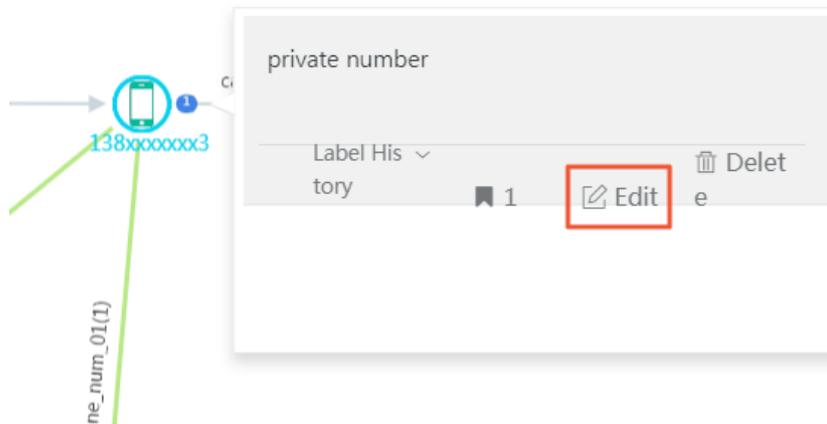
If the name, color, or visible range of a user label is unacceptable, you can edit the user label.

Prerequisites

- You have obtained the account and password with the permission to perform label-related operations.
- Other users have created user labels that are visible to you.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Open an existing analysis file or create an analysis, and add or locate a node that has user labels that are visible to you.
3. Click the label number next to the node, click a label name, and view the label details.



4. Click **Edit**, and re-set the parameters in the dialog box that appears.
For more information about the parameter settings, see [Parameters and descriptions in Add user labels](#).
5. Click **OK**.

8.12.14.7. Delete user labels

You can delete unnecessary user labels.

Prerequisites

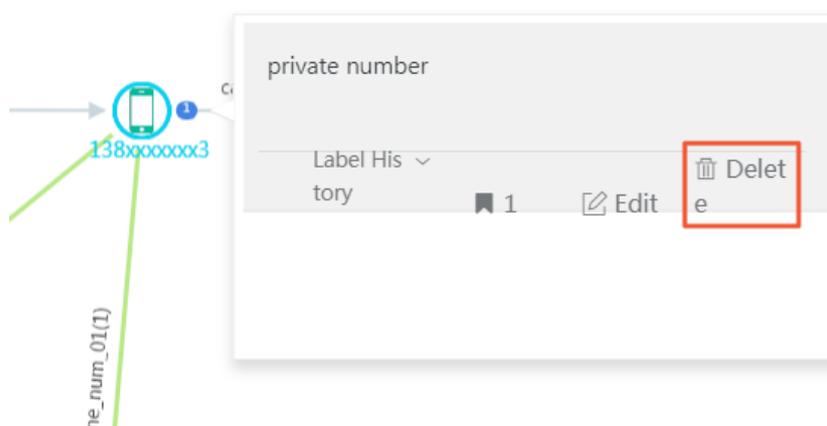
You have obtained the account and password with the permission to perform label-related operations.

Context

If the label is created by you and the number of likes is 1, you can cancel the like, and the number of likes becomes 0. When the number of likes for a label becomes 0, the system automatically deletes this label.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis file or create an analysis, and add or locate a node that has user labels that are visible to you.
3. Click the label number next to the node, and click a label name to view the detailed information about the label.



4. Click the **Delete** icon, and click **OK** in the dialog box that appears. A message is displayed, indicating that the label has been deleted successfully. If the label is created by you and currently has only one like, you can click the delete icon  to delete the like, and the number of likes becomes 0. The system will

automatically delete this label.

8.12.15. Save analysis

After the analysis is modified, you must save the modifications before you close the analysis. Graph Analytics provides a screenshot analysis function. When an analysis is saved, Graph Analytics generates a global screenshot for the analysis graph and saves the screenshot to the server.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have created a new analysis or modified an existing analysis.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Open an existing analysis file or create a new analysis.
3. (Optional) Perform multiple operations on an existing analysis or a new analysis in the graph area, such as adding nodes, setting layouts, and performing analyses.
4. The following may occur when you click the **Save** icon  in the toolbar.

Operation	Description
Save existing analyses	Saves the analysis content in the graph area by the original name and the original path.
Save new analyses	If you are creating a new analysis, you need to set File Name and Folder of the analysis in the Save Analysis dialog box that appears, and click OK to save the analysis content in the graph area.

8.12.16. Print graph areas

Analytics Workbench enables you to print the content of an analysis from different perspectives. You can export the content as a paper document, or save the content as a PNG image.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- Make sure that the system has connected to a printing device.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Open an existing analysis, or create a new analysis and have the results analyzed.
3. You can print the analysis content or save the analysis content as an image.

Print area	Operation
------------	-----------

Print area	Operation
Print the full graph	<ol style="list-style-type: none"> i. In the toolbar, choose Print > Print Full Graph . The print settings page appears. ii. Set the print parameters, and then click Print to print all the analysis content in the graph area.
Print the visible area	<ol style="list-style-type: none"> i. In the toolbar, choose Print > Print Visible Area . The print settings page appears. ii. Set the print parameters, and then click Print to print the analysis content in the visible area.
Print the selected subgraph	<ol style="list-style-type: none"> i. In the toolbar, choose Print > Print Selected . The print settings page appears. ii. Set the print parameters, and then click Print to print the analysis content of the selected nodes in the graph area.
Save the full graph as an image	In the toolbar, choose Print > Save Full Graph as Image . Save the analysis content as an image in the PNG format.

8.12.17. Share analyses

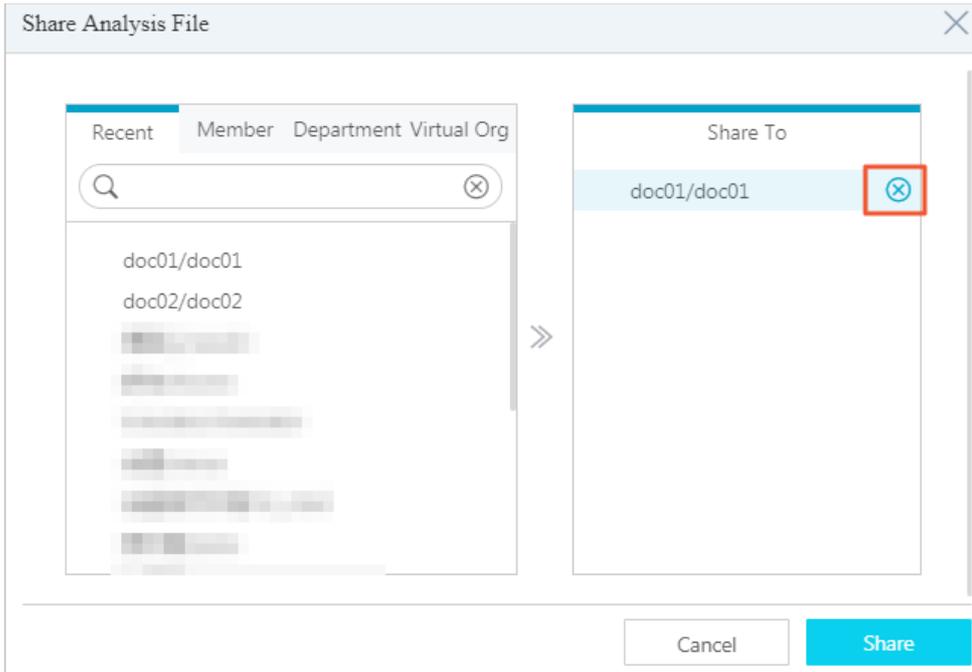
You can use Graph Analytics to share your current analysis with a specific user or group to perform a collaborative analysis. The shared users can view the shared analysis file after they log on to Analytics Workbench.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Analytics Workbench](#).
2. Open an existing analysis or create a new analysis.
3. In the toolbar, click the **Share** icon and set the parameters in the dialog box that appears.



You can select the shared members individually by using the search and positioning feature. You can also select the shared members by department.

Members selected for sharing will be displayed in the **Share To** list on the right side. When the mouse pointer is moved over the member, you can see a Delete icon. You can click the icon to delete the current member.

4. After you have specified the shared members, click **Share**, and the system informs you that you have shared the analysis successfully.

What's next

The user you shared files with can choose **File Center > Shared with Me** in the top navigation bar to view and operate on the shared analysis files.

After a member receives a shared analysis, the system automatically creates a directory with the same name as the source analysis on the **Shared with Me** page. By default, the directory has two files: the **initial file** and the **automatically merged file**.

8.12.18. Behavior chronology

8.12.18.1. Details

Details are used to display the link and event details of an object.

Prerequisites

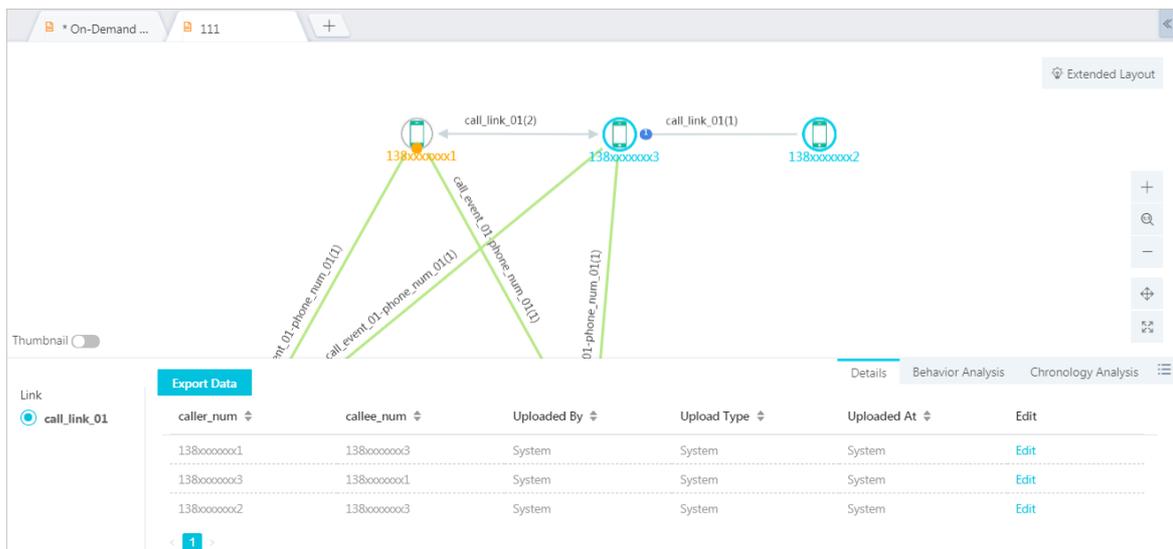
Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Analytics Workbench](#).
2. Click an analysis file to open the file in the **Graph** area.
3. You can open the **Behavior Chronology** pane by using the following methods.

Method	Procedure
Method 1	Select an object in the Graph area, and click Behavior Chronology in the lower-right corner. The Details area is displayed by default.
Method 2	Select a link in the Graph area, and the Behavior Chronology pane appears automatically. The Details area is displayed by default.

4. On the left side of the **Behavior Chronology** pane, select a link or an event to view its details.



5. (Optional) Click **Export** to export the lists of behavior details for all links to an Excel file.

8.12.18.2. Behavior analysis

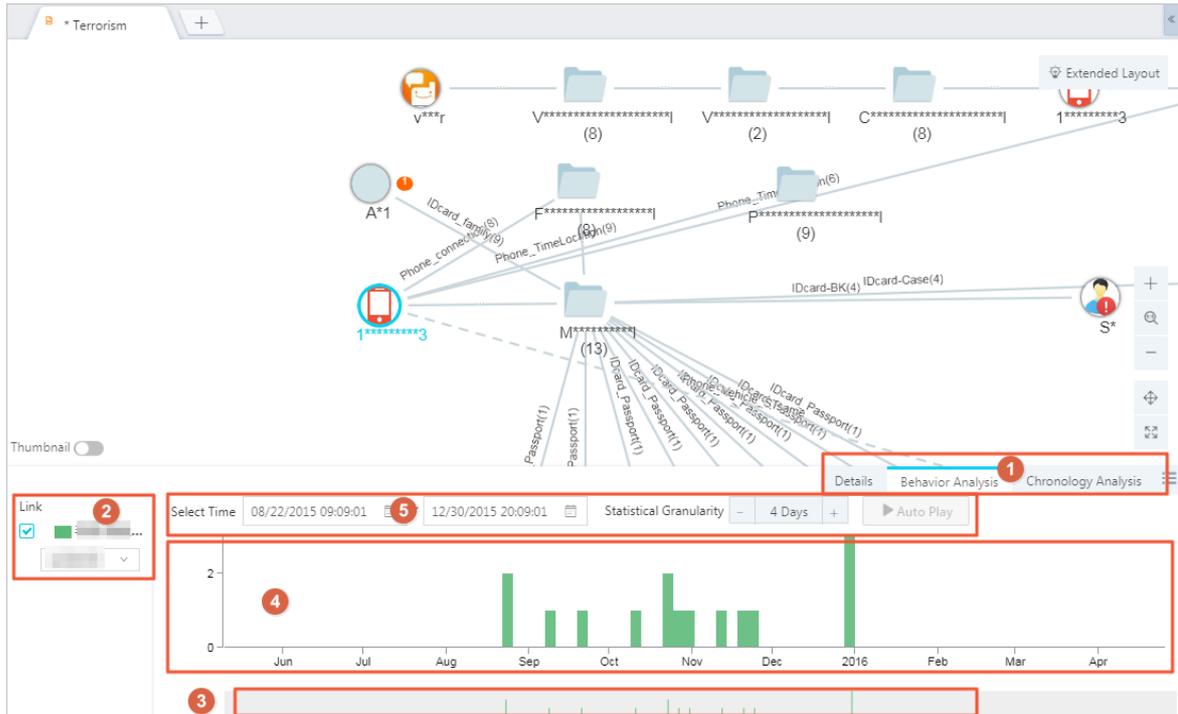
This feature allows you to display and analyze the link data with time properties on the Graph page.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click an analysis file to open the file in the **Graph** area.
3. In the **Graph** area that appears, select an object for the behavior analysis.
4. Click the **Behavior Chronology** icon in the lower-right corner. In the **Behavior Chronology** pane that appears, click the **Behavior Analysis** tab.



The behavior analysis area is described as follows.

Description of the behavior analysis area

Area	Description
Area 1	You can switch to the Behavior Analysis tab.
Area 2	You can filter the links to be analyzed.
Area 3	You can filter data using the thumbnail. You can locate the range through the thumbnail at the bottom, and select the middle part to move the thumbnail.
Area 4	<p>Move your mouse pointer to the column chart. Links that occurred at the current time point and the number of times the links have occurred are displayed.</p> <p>Click the column chart. The links and objects correlated to these links will be highlighted in the Graph area.</p> <p>Click and hold down the left mouse button and drag to select a time range. All links in the time range and objects correlated to these links will be highlighted in the Graph area.</p>
Area 5	<p>You can set the filter time range and the statistical granularity:</p> <ul style="list-style-type: none"> Set the start time and the end time for time-based filtering. Click the + icon or the - icon to increase or decrease the statistical granularity.

- For more information about how to analyze the behaviors of an object in the **Behavior Analysis** area, see [Description of the behavior analysis area](#).

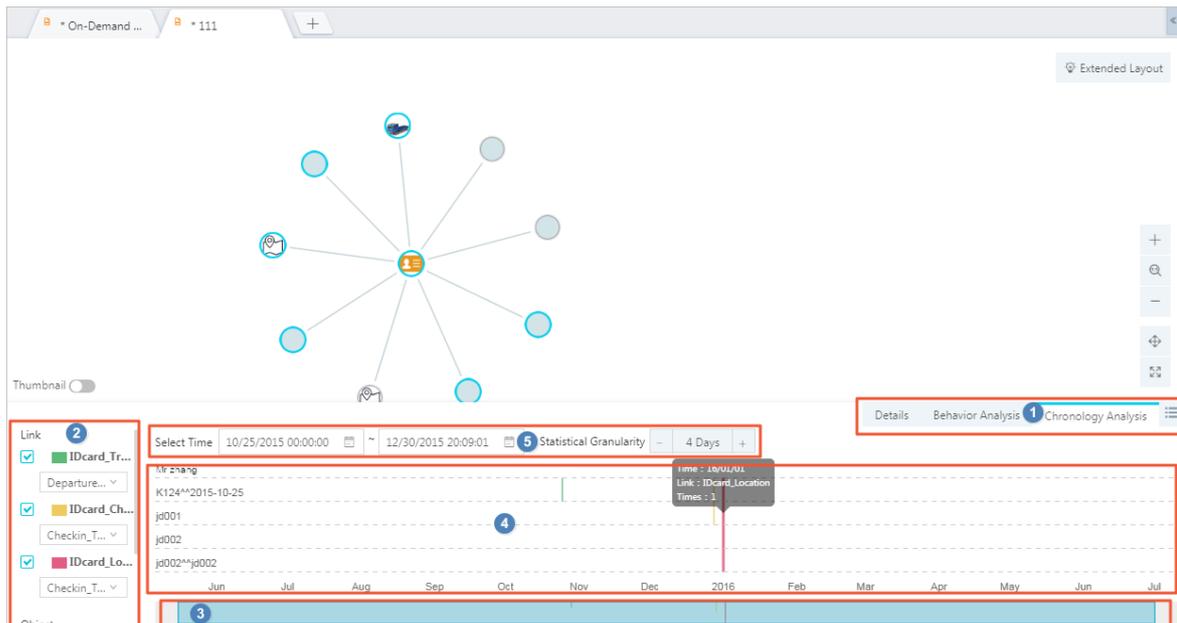
8.12.18.3. Chronology analysis

The chronology analysis shows the details of each event based on time.

Prerequisites

Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

1. Log on to [Analytics Workbench](#).
2. Click an analysis file to open the file in the **Graph** area.
3. In the **Graph** area, select the object and link to perform a chronology analysis.
4. Click the **Behavior Chronology** icon in the lower-right corner. In the **Behavior Chronology** pane that appears, click the **Chronology Analysis** tab.



By default, only five objects can be displayed for an analysis in the chronology analysis area.

The chronology analysis area is described as follows:

Description of the chronology analysis area

Area	Description
Area 1	You can switch to the Chronology Analysis tab.
Area 2	You can select a link and an object. You can analyze a maximum of five objects at the same time.
Area 3	You can filter data using the thumbnail. You can locate the range through the thumbnail at the bottom, and select the middle part to move the thumbnail.
Area 4	<p>When you move the mouse pointer to a specific chronology line, the details of the chronology appear.</p> <p>When you click the chronology line, the links and the objects involved in these links will be highlighted in the Graph area.</p> <p>Click and hold down the left mouse button and drag to select a time range. All links in the time range and objects correlated to these links will be highlighted in the Graph area.</p>

Area	Description
Area 5	You can set the filter time range and the statistical granularity: <ul style="list-style-type: none"> Set the start time and the end time for time-based filtering. Click the + icon or the - icon to increase or decrease the statistical granularity.

- For more information about how to analyze the behaviors of an object in the **Behavior Analysis** area, see [Description of the chronology analysis area](#).

8.12.19. Property statistics

8.12.19.1. Details

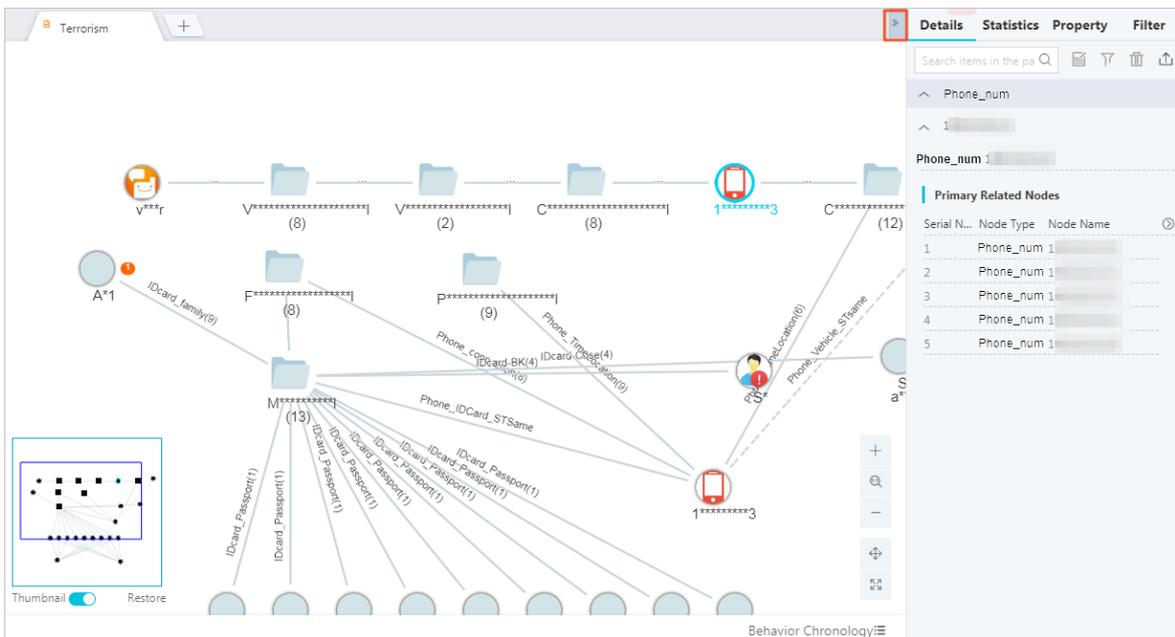
Typically, when you perform an analysis in Graph Analytics, a large number of objects will be involved. You may need to highlight the objects that are important to you. The properties and statistics area displays data with multiple properties. You can highlight the key objects on the Graph page.

Prerequisites

You have obtained an account and a password with the permission to perform network analyses.

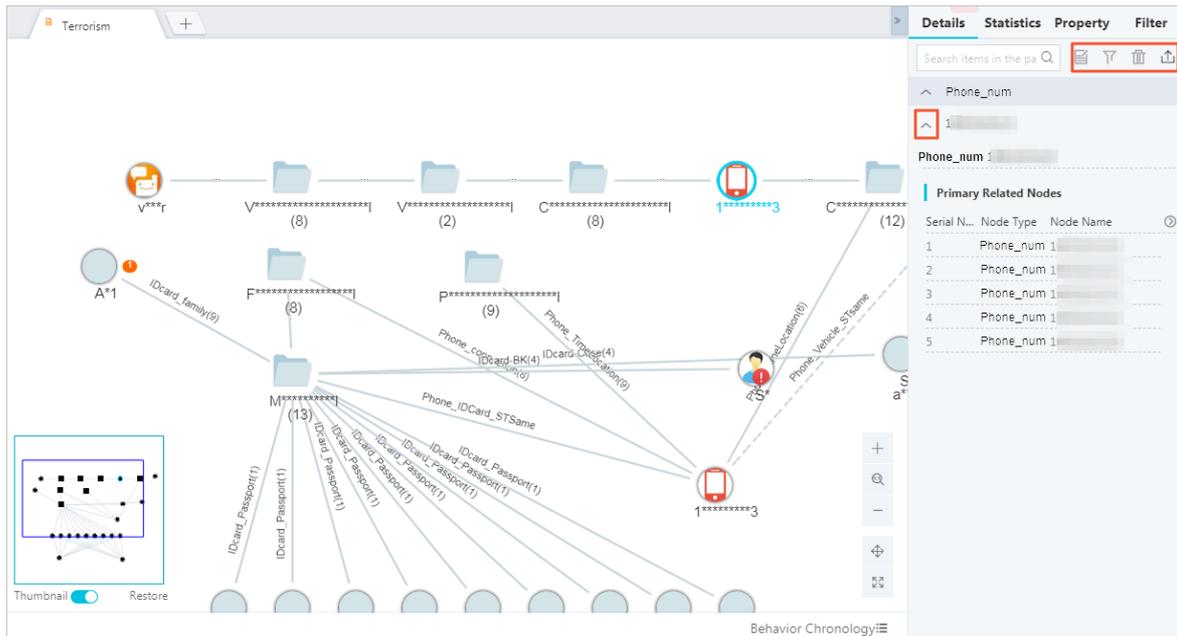
Procedure

- Log on to [Analytics Workbench](#).
- Click an analysis file to open the file in the **Graph** area.
- Select one or more objects or links in the **Graph** area.
- Click the  icon in the right side of the graph area to display the properties and statistics. By default, the **Details** tab is displayed.



The **Details** tab displays the basic information of the selected object, including the object type, object icon or avatar, object ID, object properties, and the correlated nodes.

- You can perform the following operations in the **Details** tab.



Operation	Procedure
Highlight key objects	When you select objects in the right-side pane, the objects, links, and events in the graph area are masked, except for the selected objects. You can press the Control key to select multiple objects.
View object details	When you select nodes or links in the main graph area, the details page on the right side shows all the property information of the selected nodes and links. You can press the Control key to select multiple nodes or links. If you select a link in the graph area, the details of the objects involved in the link are displayed.
Add text notes	Add text notes or labels. For more information, see Add user labels .
Select	Select the node in the graph area.
Delete	Delete the selected object in the graph area.
Export	Export the information of the selected object to an Excel file.

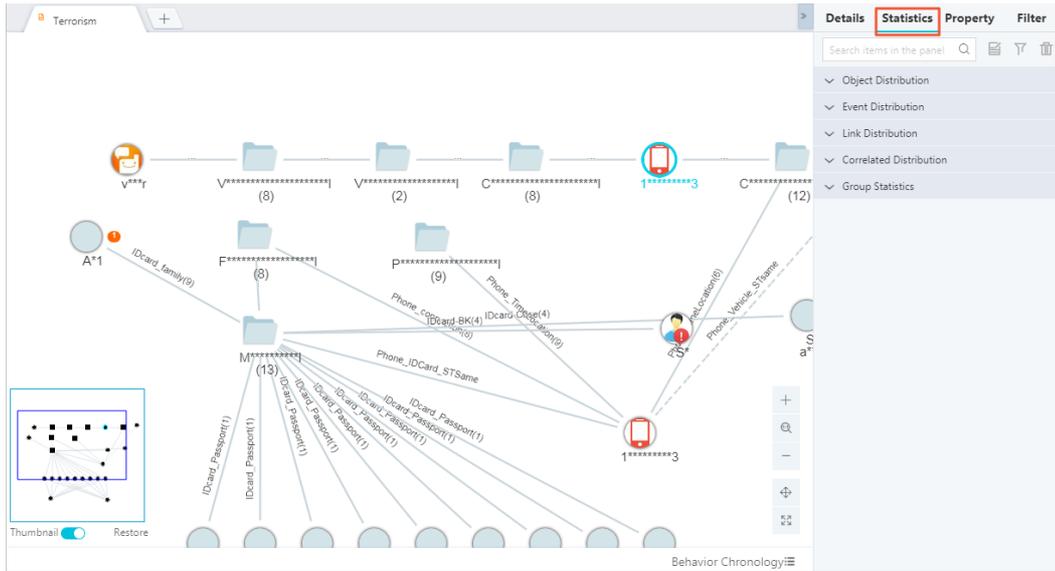
8.12.19.2. Statistics

The Statistics tab displays the statistics of the selected objects or links.

Procedure

1. [Log on to Analytics Workbench](#).
2. Select one or more nodes or links in the graph area.
3. Click the right arrow in the right side of the graph area to display the properties and statistics.
4. Switch to the **Statistics** tab, as shown in [Statistics tab](#).

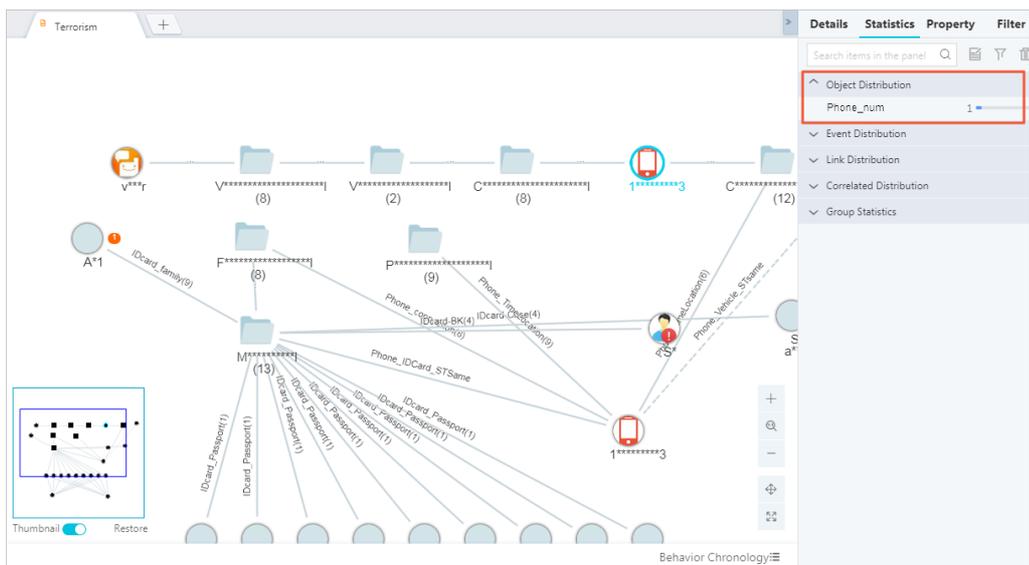
Statistics tab



5. Click the up or down arrow to hide or show the properties of the selected object or link.

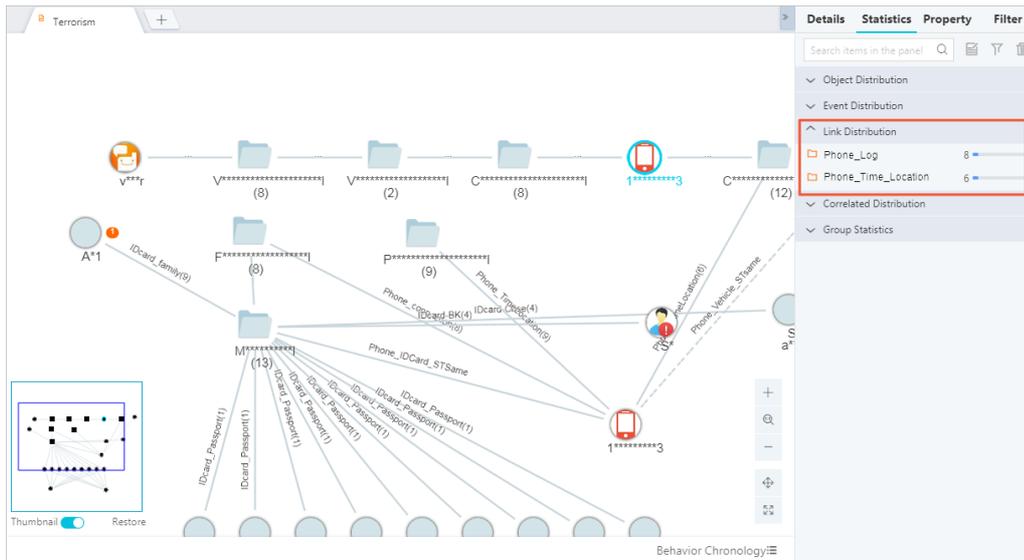
- View the object distributions, as shown in [Object distributions](#).

Object distributions



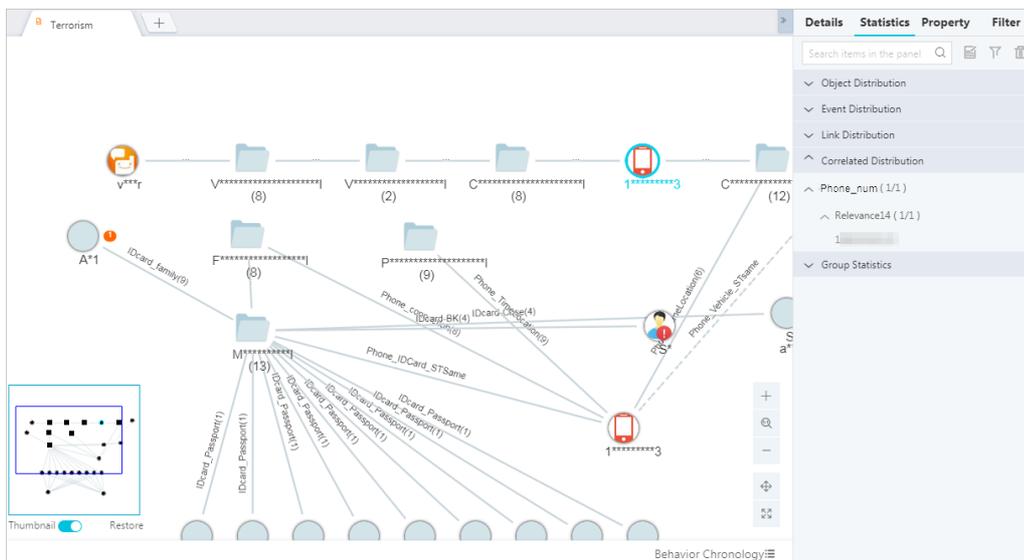
- View the link distributions, as shown in [Link distributions](#).

Link distributions



- View correlation distributions, as shown in **Correlation distributions**.

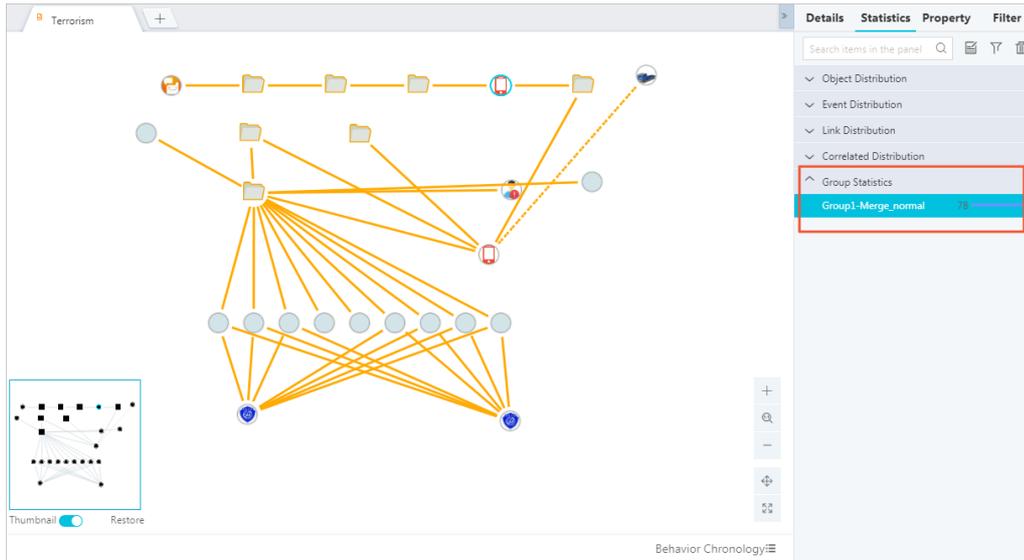
Correlation distributions



- View the group analysis, as shown in **Group analysis**.

Group statistics allows you to analyze group distributions in Graph Analytics. Group statistics is typically used to perform a group analysis. After you enter multiple nodes, Graph Analytics analyzes the interrelations between these nodes, provides the analysis results, and displays the group distribution. A group consists of multiple object nodes, with any two object nodes connected topologically. Nodes within a merged node are connected topologically.

Group analysis



- In group statistics, all isolated nodes form a group are called isolated nodes.
- The list in group statistics displays the number of nodes and the labels of the nodes that have the highest correlated degree in each group, excluding the group of isolated nodes. Group of isolated nodes is displayed at the top of the list, while other groups are listed in a descending order according to the number of nodes contained in the group.

Note Group statistics cover all nodes in Graph Analytics, regardless of your selection of nodes during the analysis process.

On the Statistics page, you can perform the following operations for the selected content:

- Text Note: You can add text notes. For more information, see [Add user labels](#).
- Selected: You can select the node in the graph area.
- Delete: You can delete the selected object in the graph area.

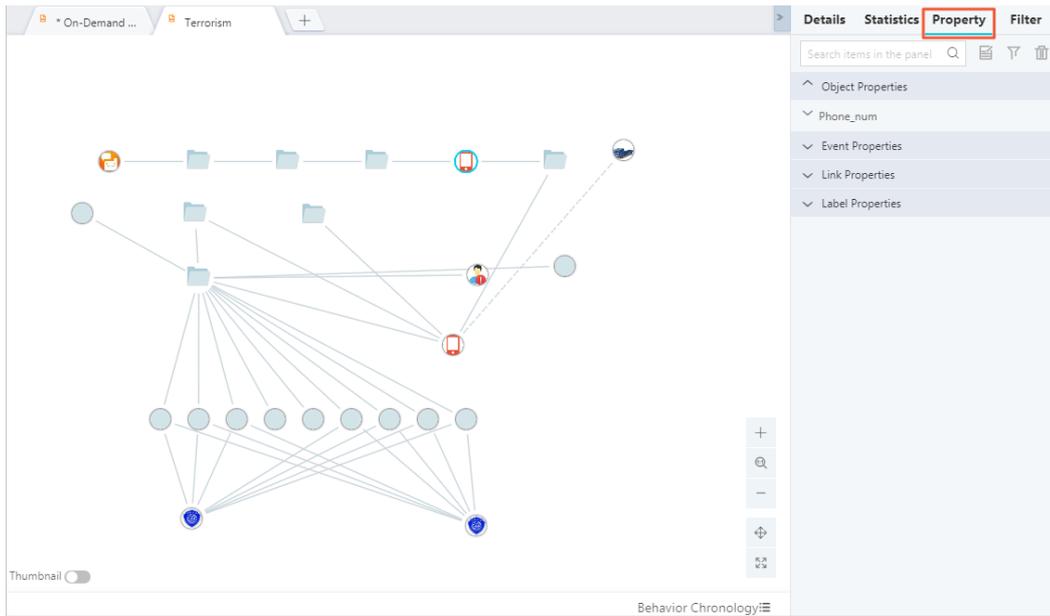
8.12.19.3. Property information

The Property tab in the right-side pane of the Graph page displays the object or link information and the label information of the selected objects or links.

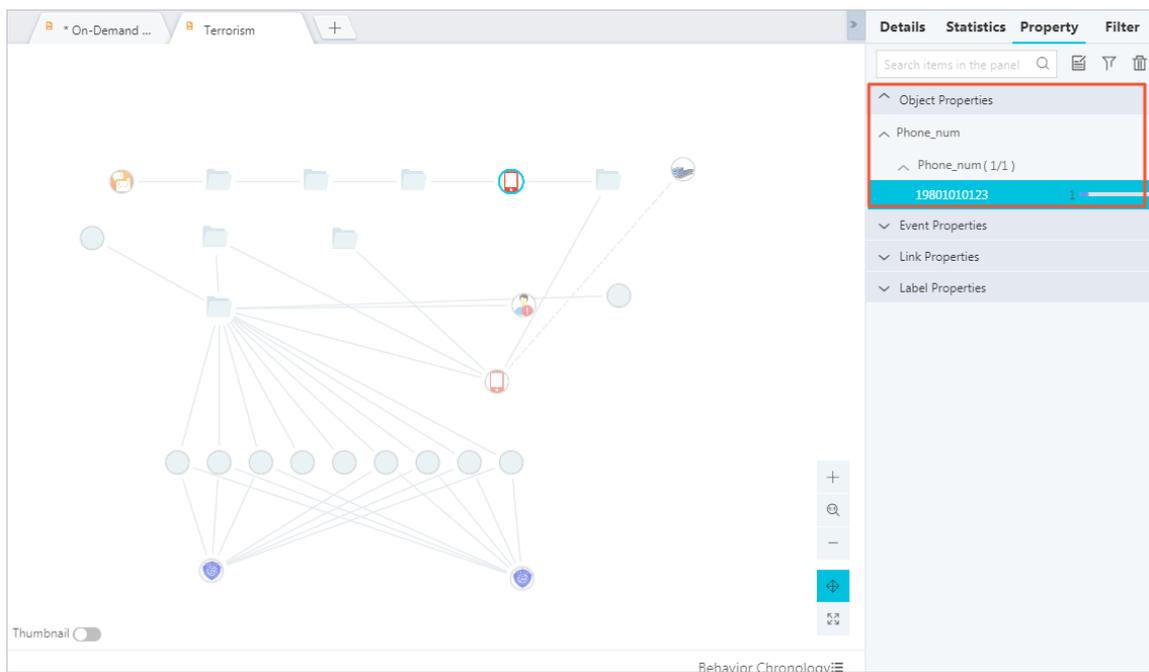
Procedure

1. [Log on to Analytics Workbench](#).
2. Select one or more nodes or links in the graph area.
3. Click the right arrow in the right side of the graph area to display the properties and statistics.
4. Click the **Property** tab. The Property tab appears, as shown in [Property tab](#).

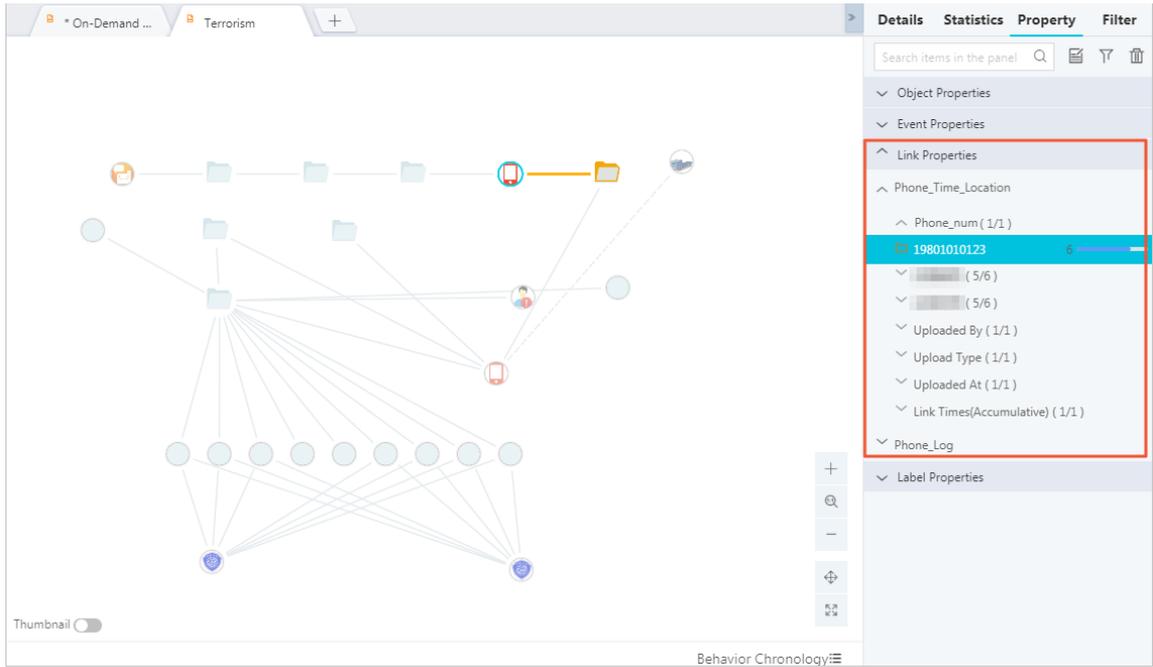
Property tab



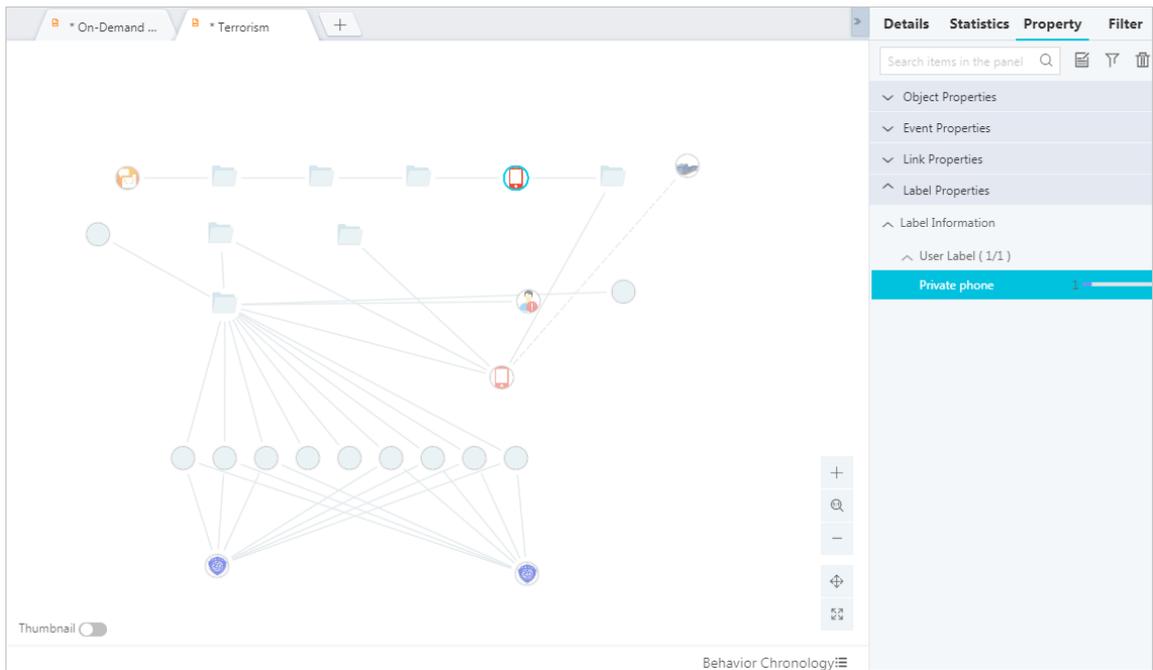
- 5. Click the up or down arrow to hide or show the properties of the selected object or link.
 - o View object properties.



- o View event properties.
- o View link properties.



- o View label properties.



On the Property tab, you can perform the following operations for the selected content:

- o Text Note: You can add text notes. For more information, see [Add user labels](#).
- o Selected: You can select the node in the graph area.
- o Delete: You can delete the selected object in the graph area.

8.12.19.4. Secondary filtering

You can use the secondary filtering feature to filter objects, links, or events in the canvas. In a complex analysis, you can filter out irrelevant objects, links, or events to keep the content simple and concise.

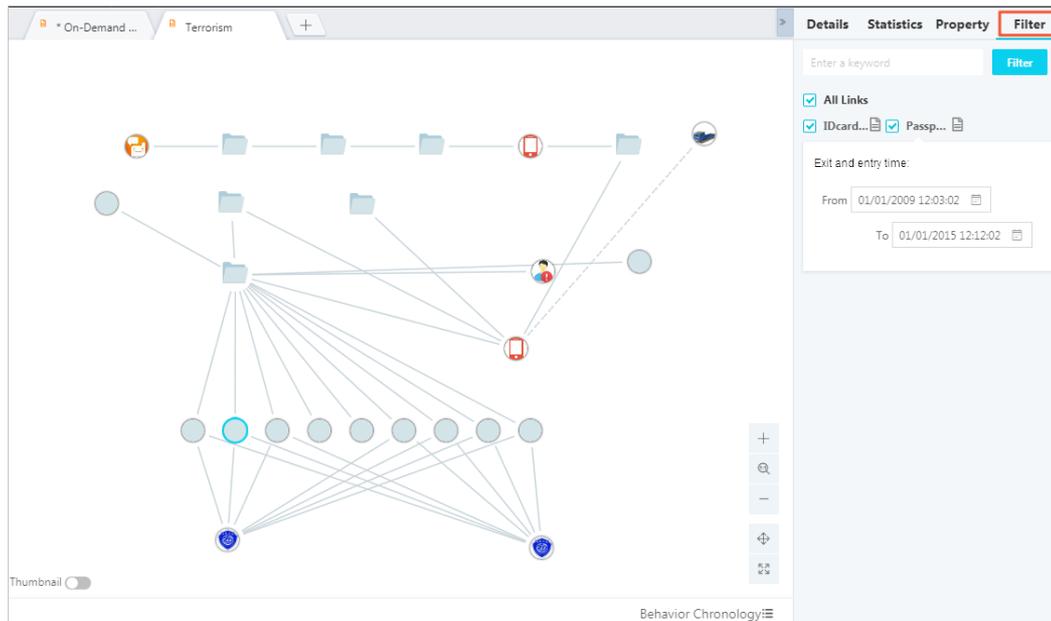
Context

In many service scenarios where data processing is complex, the canvas analysis may become very complex after a few analysis extensions. It is essential to perform secondary filtering before you make further analyses and judgments.

Procedure

1. [Log on to Analytics Workbench.](#)
2. Click the arrow-pointing-right icon in the right of the graph area to show the properties and statistics.
3. Click the **Filter** tab, and switch to the **Filter** page, as shown in [Entry to the secondary filtering.](#)

Entry to the secondary filtering



4. Enter a keyword to be filtered.

Related parameters are described as follows:

- Time type: maximum and minimum values of the time property. You can adjust the value range, for example, departure time.
- Numeric type: maximum and minimum numerical values. You can adjust the value range, for example, age.
- Dictionary type: enumerated values. You can delete some of the enumerated values.
- Character string: used for searching. Fuzzy search is supported.

8.13. File Center

8.13.1. View and manage all analyses

You can see all the personal analysis files and shared analysis files of the current user. The files are arranged in the order of creation time.

Prerequisites

The current user has created and saved analysis files, or has received shared analysis files.

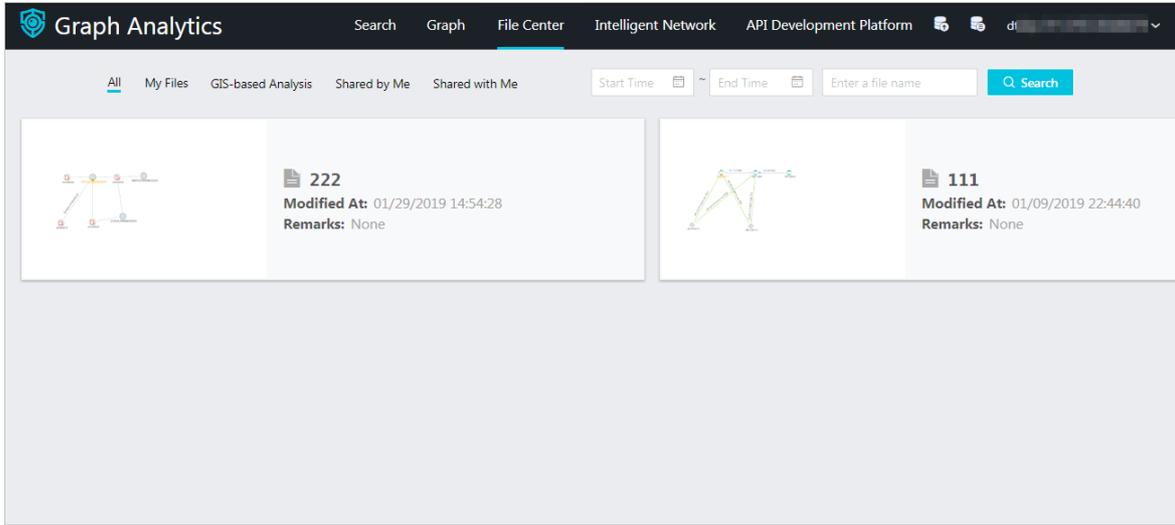
Procedures

1. [Log on to Analytics Workbench.](#)

2. In the top navigation bar, click **File Center**, and then click the **All** tab. The **All** tab appears.

The **All** page displays all the shared files and personal files.

When you handle a large number of analysis files, you can use the Search feature to find the target files.



3. On the **All** page, you can perform the following operations on each analysis:

Method	Description
Open an analysis file	Double-click an analysis file to open the file in the graph area.
Delete an analysis file	Select an analysis that you created or a shared file that you received, click the Delete icon in the lower-left corner, and click Delete in the dialog box that appears.
Rename an analysis file	<p>Note This operation is only valid for analysis files created by you.</p> <p>Select an analysis, and click the Rename icon in the lower-left corner. In the dialog box that appears, enter a new name and click OK.</p>
Change sharing permissions	<p>Note This operation is only valid for files that have been shared by the current user.</p> <p>Select a shared file, and click the Change Sharing Permissions icon in the lower-left corner. In the dialog box that appears, reset the shared members and click Share.</p>

8.13.2. View and manage your files

You can view all your personal folders and analysis files in the order of creation time. You can perform operations on the folders and analysis files, such as add, delete, view, and modify.

Prerequisites

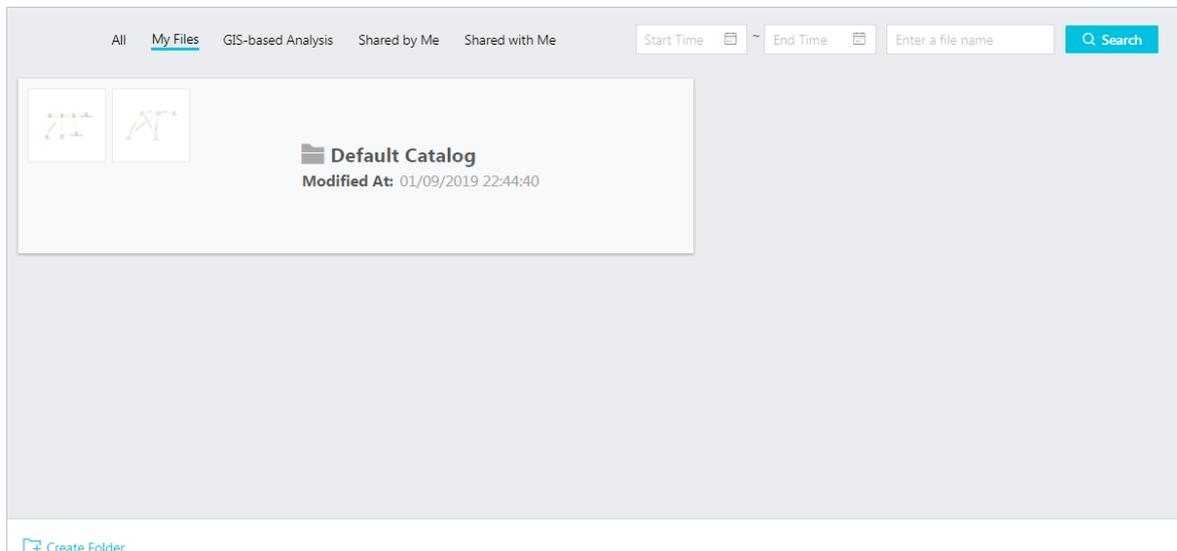
Make sure that you have created and saved an analysis file.

Procedure

1. [Log on to Analytics Workbench.](#)
2. In the top navigation bar, click **File Center** and select the **My Files** tab. The **My Files** page appears.

The **My Files** page displays all folders that contain analysis files that have been saved.

You can use the Search feature to handle a large number of folders or analysis files.



3. On the **My Files** page, you can perform the following operations.

Operation	Description
Create a folder	Click Create Folder in the lower-left corner of the page. In the dialog box that appears, enter a folder name, and then click OK .
Rename a folder	Select the folder to be renamed, and click Rename in the lower-left corner of the page. In the dialog box that appears, enter a new name, and then click OK .
Delete a folder	Select the folder to be deleted, and click Delete in the lower-left corner of the page. In the message that appears, click Delete .
Rename an analysis file	<ol style="list-style-type: none"> Double-click a folder to open the folder. Select an analysis file to be renamed, and click Rename in the lower-left corner of the page. In the dialog box that appears, rename the analysis file, and then click OK.
Delete an analysis file	<ol style="list-style-type: none"> Double-click a folder to open the folder. Select an analysis file to be deleted, and click Delete in the lower-left corner of the page. In the message that appears, click Delete.
Move an analysis file	<ol style="list-style-type: none"> Double-click a folder to open the folder. Select the analysis file to be moved, and click Move in the lower-left corner of the page. In the dialog box that appears, select the target folder, and click OK.

Operation	Description
Share an analysis file	<ol style="list-style-type: none"> i. Double-click a folder to open the folder. ii. Select the analysis file to be shared, and click Share in the lower-left corner of the page. In the dialog box that appears, select the members you want to share the file with, and click Share.

8.13.3. My shared items

8.13.3.1. Overview

Graph Analytics allows you to share personal analyses with others. You can share personal ideas and experiences with other users and combine their intelligence and experiences to achieve collaboration and build a better team.

The initiator and the collaborators are the main roles in the collaboration and sharing process. A collaboration proceeds as follows.



After a member receives a shared analysis, the system automatically creates a directory with the same name as the source analysis on the **Shared with Me** page. By default, the directory has two files: the **initial file** and the **automatically merged file**.

Graph Analytics allows you to manage shared files, including deleting files, renaming files, and managing version history.

Note Only initiators have the permissions to delete and rename shared files.

8.13.3.2. View and manage shared files

The Shared by Me page displays all the files shared by the current user in the order of time when the files were created. You can delete folders and change sharing permissions. You can also merge shared files and delete version files.

Prerequisites

You have shared an analysis file. For more information, see [Share analyses](#).

Context

On the Shared by Me page, you can perform the following operations.

Operation	Description
Delete a shared folder	After the sharer deletes the shared folder, the files will also be deleted from members to which the files are shared.
Modify sharing permissions	After the sharer modifies the sharing permissions, the permissions of the original sharing members are revoked, and the shared files are also deleted.

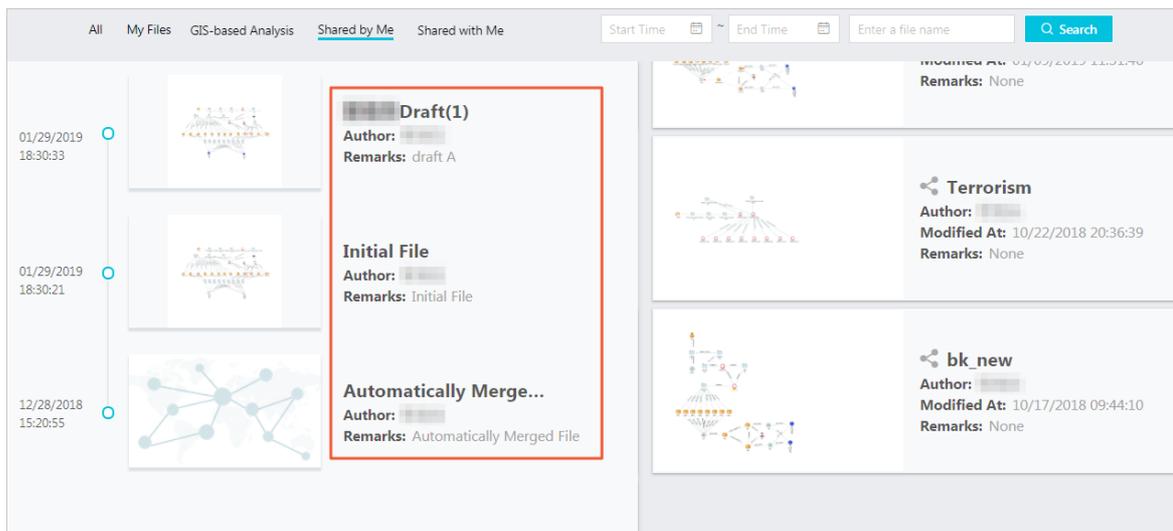
Operation	Description
Delete a shared file	 Note You cannot delete the initial file or the automatically merged file that exist by default.
Manually merge files	You can merge multiple published versions in a shared folder to form a new version. Only published file versions can be merged. After the files are merged, a manually merged file is generated. If the published versions are different, use the union of the versions.

Procedure

1. [Log on to Analytics Workbench.](#)
2. In the top navigation bar, click **File Center**, and then click the **Shared by Me** tab. The **Shared by Me** page appears.

The **Shared by Me** page displays all folders that the current user has shared. Each folder is a shared item. It contains multiple draft files, one initial file, one automatically merged file, multiple manually merged files, and multiple published version files.

When a large number of folders or files are displayed, you can use the search function to query the target.



3. On the **Shared by Me** page, you can perform the following operations.

Operation	Description
Delete a shared folder	Select a shared folder, click the Delete icon in the lower-left corner, and click Delete in the dialog box that appears.
Modify sharing permissions	Select a shared folder, and click the Change Sharing Permissions icon in the lower-left corner. In the dialog box that appears, reselect the members to whom you want to share the folder, and click Share .

Operation	Description
Delete a shared file version	<ul style="list-style-type: none"> i. Double-click the shared folder, and select a file. You can delete draft files, manually merged files, and published version files. ii. Click the Delete icon in the lower-left corner, and click Delete in the dialog box that appears.
Manually merge files	Select two or more published version files, and click the Merge Files icon in the lower-left corner. In the dialog box that appears, enter the description and click OK .

8.13.3.3. Edit a shared file

The sharer and shared members can edit shared files, including initial files and automatically merged files. After a file is edited and saved, a draft file will be generated. Only the current user can view the draft.

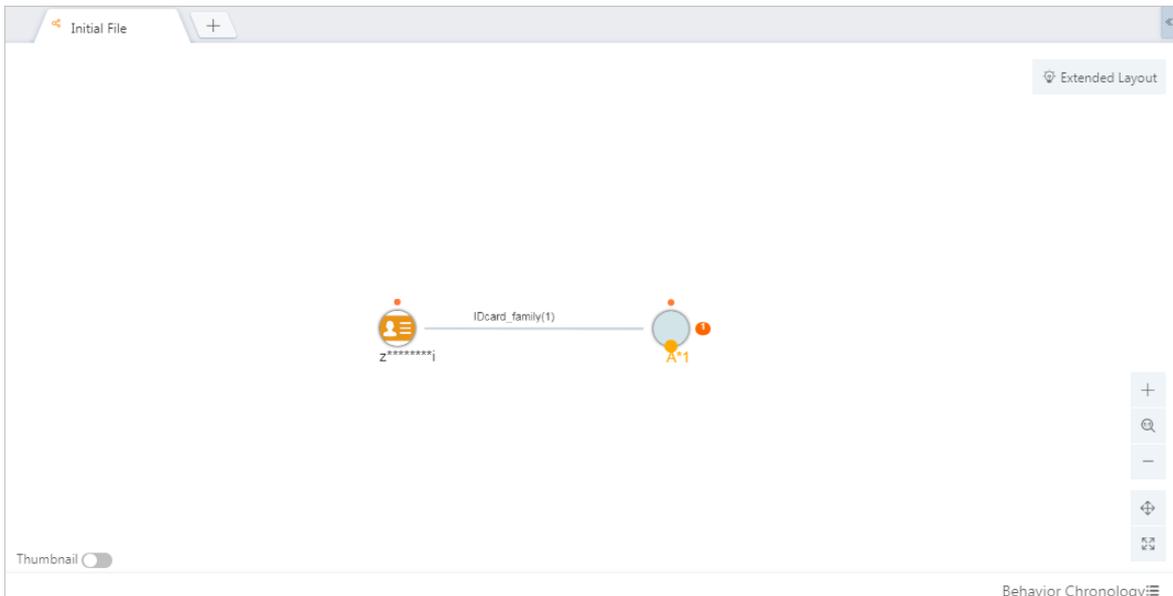
Prerequisites

You have shared an analysis file. For more information, see [Share analyses](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **File Center**, and then click the **Shared by Me** tab. The **Shared by Me** page appears.
3. Double-click a shared folder, and then double-click a shared file in the folder, such as the initial file or the automatically merged file, to open the shared file in the graph area.

Each object node in the initial file has a red dot above it to identify itself as the initial node. Subsequently extended object nodes do not have this red dot.

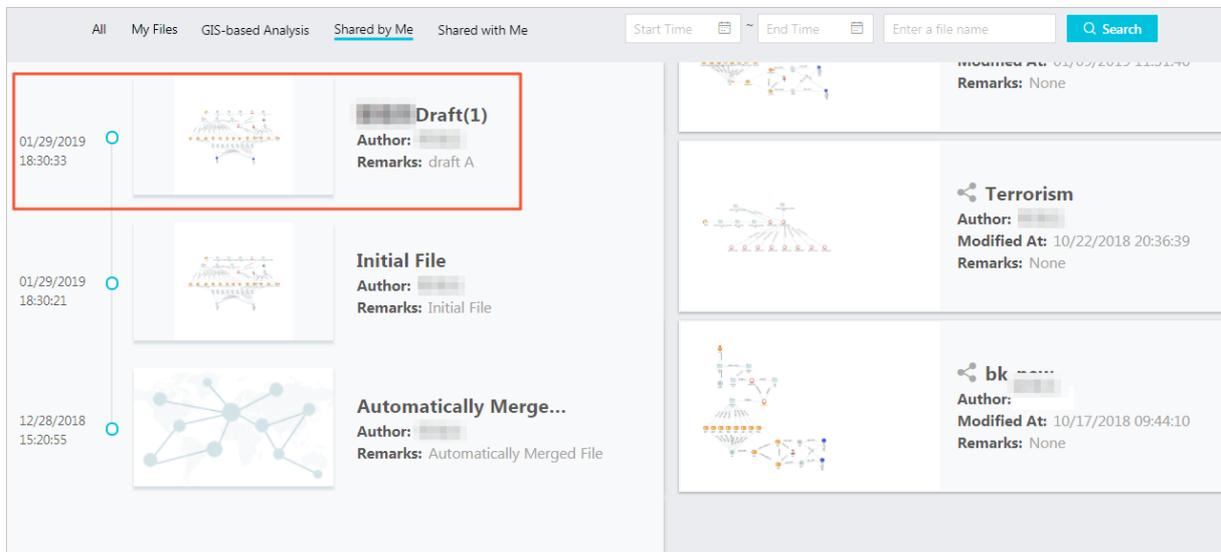


4. Perform analysis or layout operations on the shared file as needed.
5. After the analysis results are shown, click the **Save** icon in the toolbar. In the dialog box that appears, enter the Draft Statement and click **Save**.

Draft Statement: Enter the draft statement. The statement must be 1 to 200 characters in length.

Result

After a draft of a shared file is saved, the draft file is displayed in the directory of the shared file. However, the draft is visible only to the current user.



- Location: above the **initial file**.
- Name format: username of the current user + draft + (number). For example, test 123draft(1).
- Author: the user who saved this draft.
- Description: displayed below the draft file name.
- Time: the time when the draft was created is displayed on the left side of the file.

8.13.3.4. Publish a version

You can edit a shared analysis file and publish a new version. The published analysis file can be viewed by all users with whom the file is shared. However, only the sharer and the publisher can delete the file.

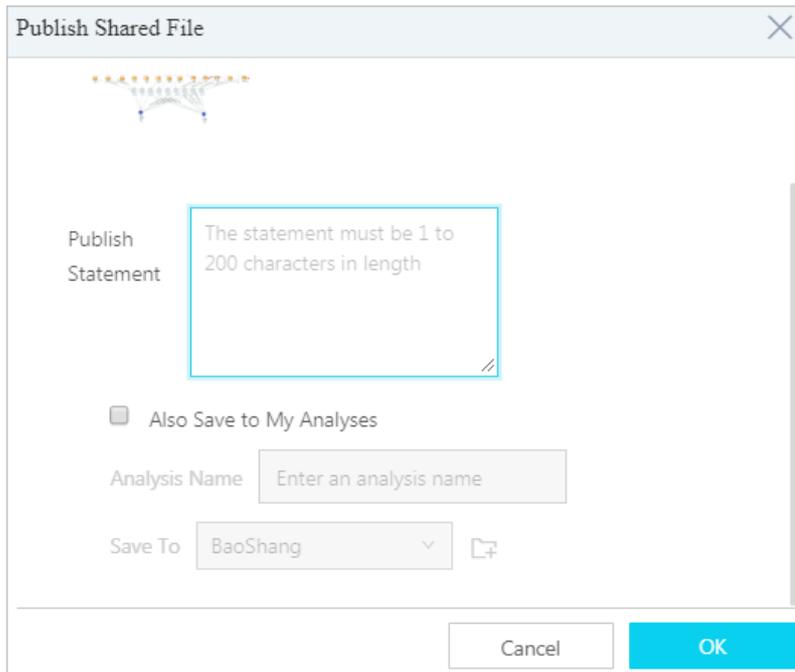
Prerequisites

You have shared an analysis file. For more information, see [Share analyses](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **File Center**, and then click the **Shared by Me** tab. The **Shared by Me** page appears.
3. Double-click a shared folder, and double-click a shared file in the folder, such as the initial file or the automatically merged file, to open the shared file in the graph area. Perform analysis or layout operations on the shared file as needed.
4. After the analysis results are shown, click the **Publish** icon in the toolbar and set the parameters in the dialog box that appears.

The parameters are described in [Publish parameters](#).



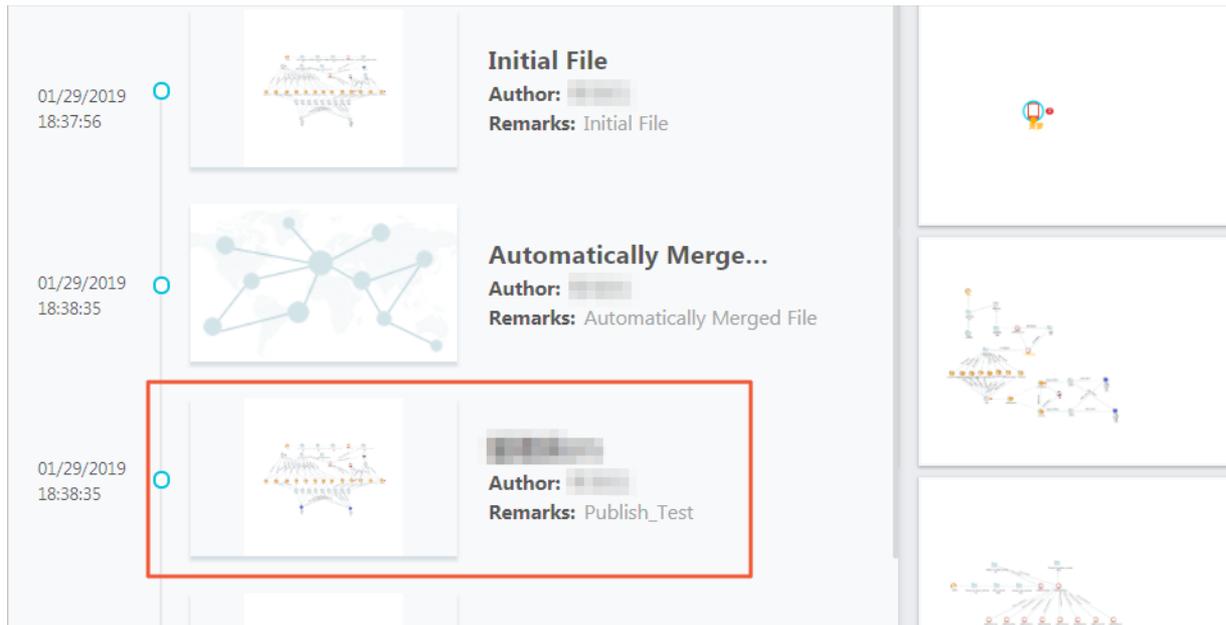
Publish parameters

Parameter	Description
Publish Statement	Enter the publish statement. The statement must be 1 to 200 characters in length.
Also Save to My Files	After you select this checkbox, the analysis file is displayed on the My Files tab page. You must enter the analysis name and the directory of the analysis file.

5. Click **OK**.

Result

After a new version of a shared analysis is published, the published version file is displayed in the directory of the shared analysis file.



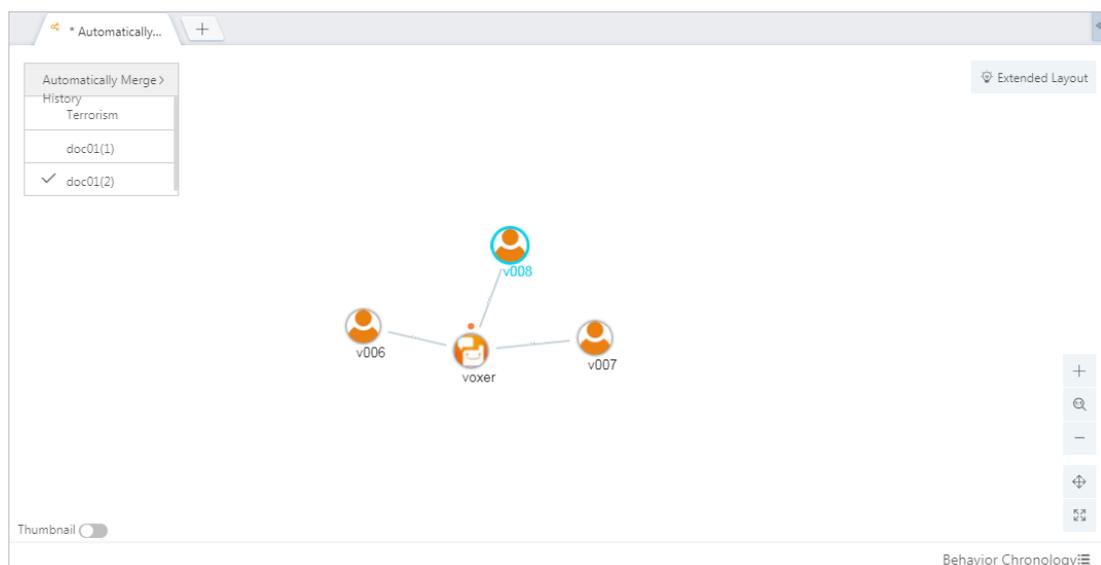
- Location: below the automatically merged file or the manually merged files.
- Name format: username + (version number). For example, test123(1).
- Author: the user who published the file.
- Description: displayed below the file name.
- Time: the time when the file was generated is displayed on the left side of the file.

8.13.3.5. Automatically merge files

When a new version is published, both the shared members and the sharer can use the auto merge feature to merge the new version with the earliest version (the initial file).

After a new version is published, the automatically merged file is updated, as shown in [Automatically merge files](#).

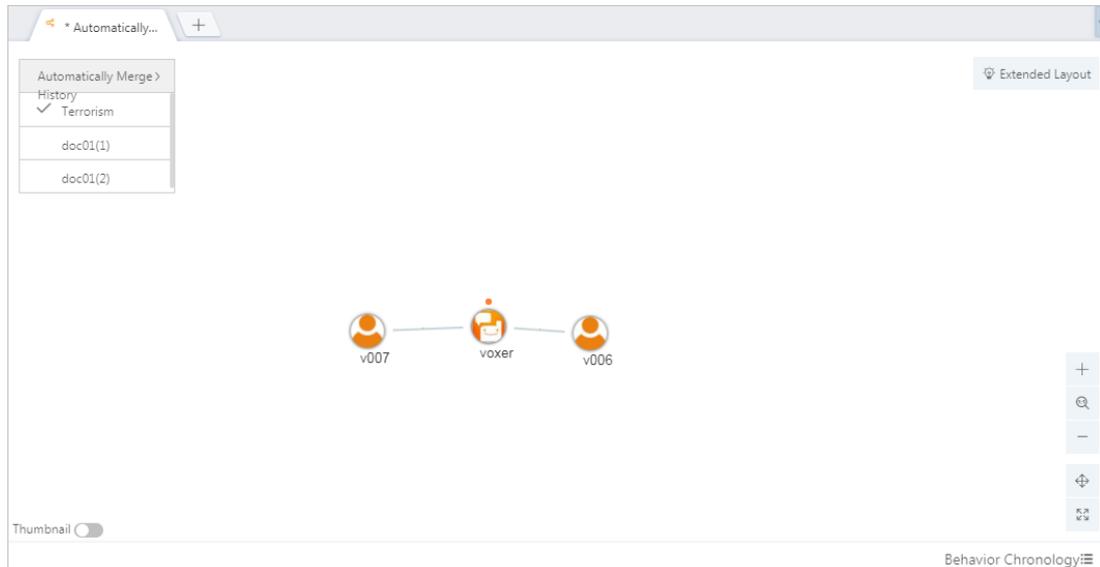
Automatically merge files



The versions are listed in the upper-left corner of the graph area. By default, the file of the latest version is displayed.

You can select an earlier version. Select a version in the **Auto Merge History** list to view the file of an earlier version, as shown in [Select an earlier version](#).

Select an earlier version



8.13.4. View and manage received shared files

You can view and modify the analysis files shared by other users on the Shared with Me page. However, you cannot delete these files.

Prerequisites

You have received a shared file.

Context

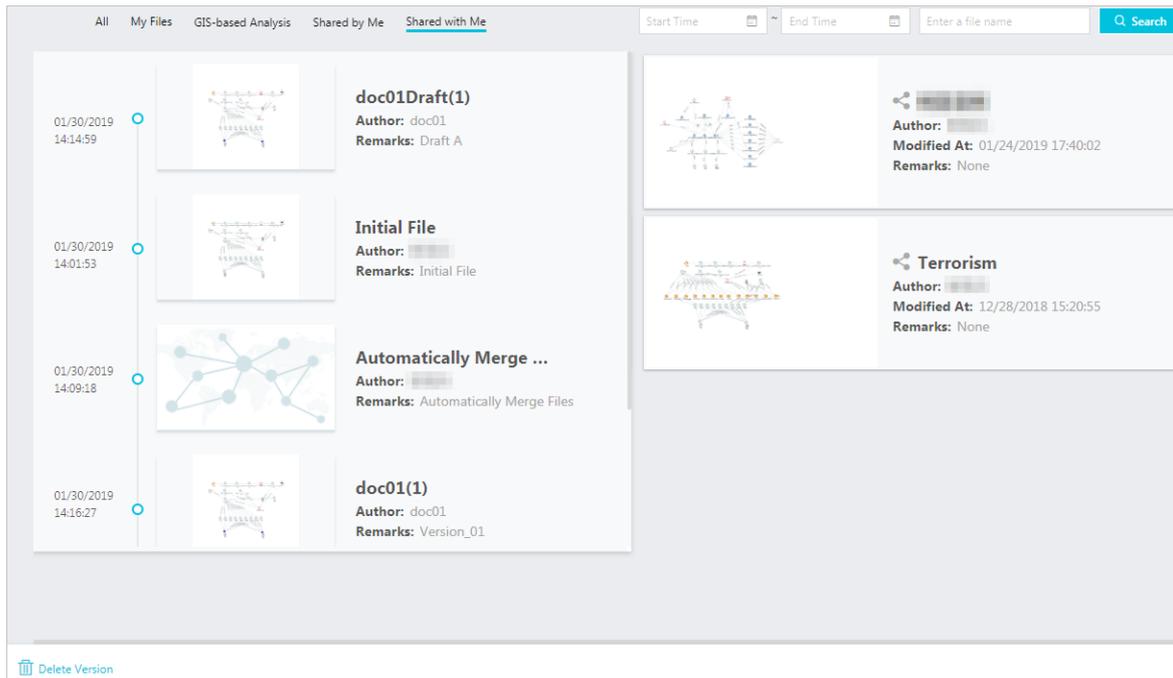
You can edit shared files, delete drafts, and publish new versions on the **Shared with Me** page.

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **File Center**, and then click the **Shared with Me** tab. The **Shared with Me** page appears.

The **Shared with Me** page displays all shared folders received by the current user. Each folder is a shared item. It contains multiple draft files, one initial file, one automatically merged file, multiple manually merged files, and multiple published version files.

When a large number of folders or files are displayed, you can use the search function to query the target.



3. On the **Shared with Me** page, you can perform the following operations.

Operation	Description
Edit a shared file	Double-click an initial file, an automatically merged file, or a published version file to open the file in the graph area. Edit and save the file, and a draft file is generated.
Delete a draft	Select the draft to be deleted, click the Delete Version icon in the lower-left corner, and click Delete in the dialog box that appears.
Publish a version	See Publish a version .

8.14. Intelligent Network

8.14.1. Intelligent Network overview

This topic describes related concepts of Intelligent Network and how to use Intelligent Network.

Concepts

Intelligent Network allows you to query subgraphs with the same graph structure as a task specified in a predefined pattern.

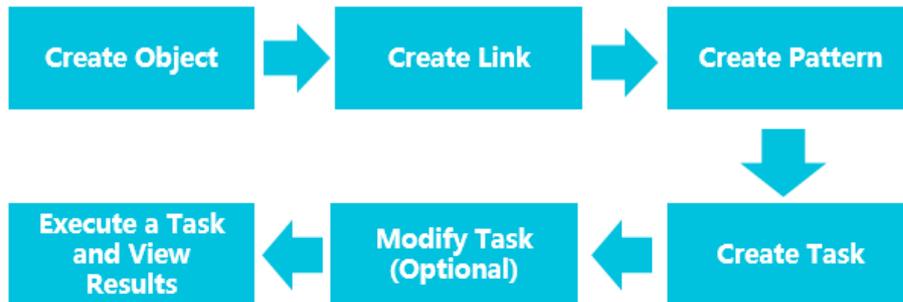
Intelligent Network involves the following concepts:

- **Pattern:** the relationship graph structure model that is predefined in Intelligent Network. Patterns are divided into private patterns and public patterns.
 - Private pattern: Only administrators and creators can use private patterns to create private tasks. Private patterns can be set to public patterns, but this is an irreversible operation.
 - Public pattern: All users can use public patterns to create public or private tasks. Public patterns cannot be set to private patterns.
- **Task:** created based on the pattern and used to query data with the same graph structure as the task in

the data source. After a task is created based on a pattern, the task has the same settings as the pattern. You can modify the graph structure and filter conditions of the task. Tasks are divided into private tasks and public tasks.

- Private task: Only administrators and creators can use private tasks. Private tasks created based on public patterns can be set to public tasks, but this is an irreversible operation.
- Public tasks: All users can use public tasks. Public tasks cannot be set to private tasks.

Procedure to use Intelligent Network



8.14.2. Patterns

8.14.2.1. Create patterns

A pattern is the relationship graph structure model predefined in Intelligent Network. It is the basis of creating a task. Before you use the intelligent network to query a sub-graph, you must use related objects and links to predefine a pattern.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have created an object and a first-degree link correlated with the object. For more information about how to create an object and a first-degree link, see [Create an object](#) and [Create a first-degree link](#).

Context

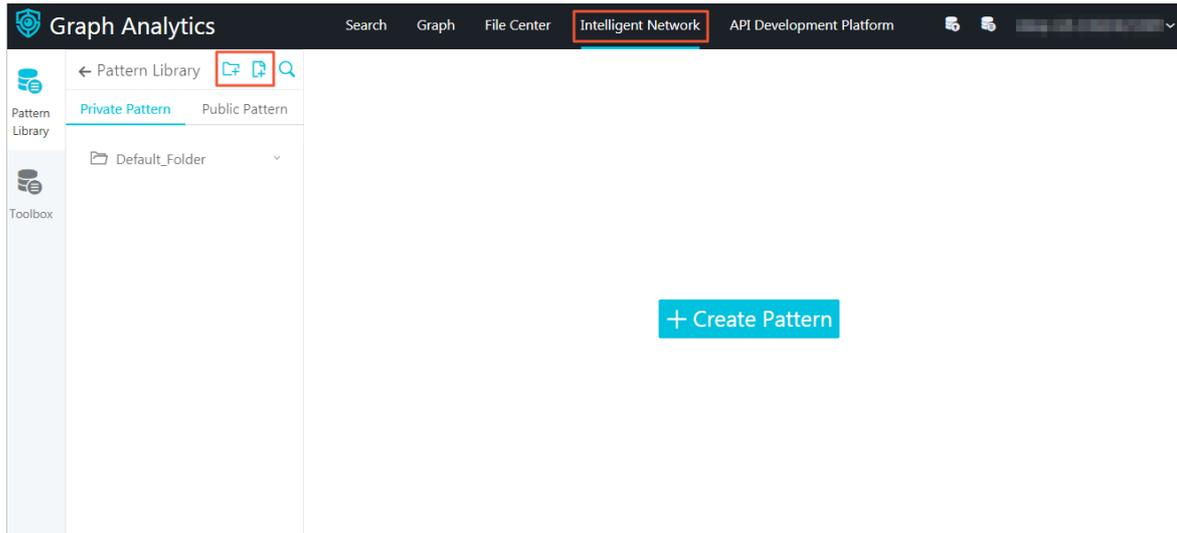
You can set the filter conditions in the **Property** tab and the **Global Conditions** tab based on your requirements.

- **Property** displays the properties of objects and links that can be used as query conditions. The **Conditional Query** properties and properties in **Accumulative Statistics Settings** will be displayed on the Property tab page.
- **Global Conditions** are based on the number and time type properties in a link. Global conditions take effect on the entire pattern. To configure **Global Conditions**, you must name the relevant number or time type properties on the **Property** tab.

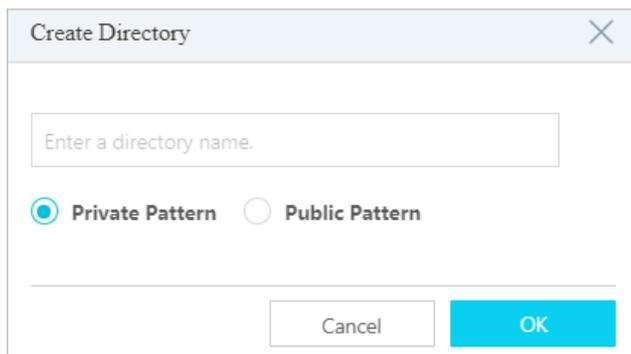
The following example shows a transfer pattern (A>B>C>A). The global condition is that the transfer from A to B is made earlier than the transfer from B to C.

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.



3. You can create a new folder to save the new pattern separately.
 - i. In the **Pattern Library** navigation pane, click the  icon (**Create Folder**).
 - ii. Specify the **Directory Name** and the pattern type as needed.

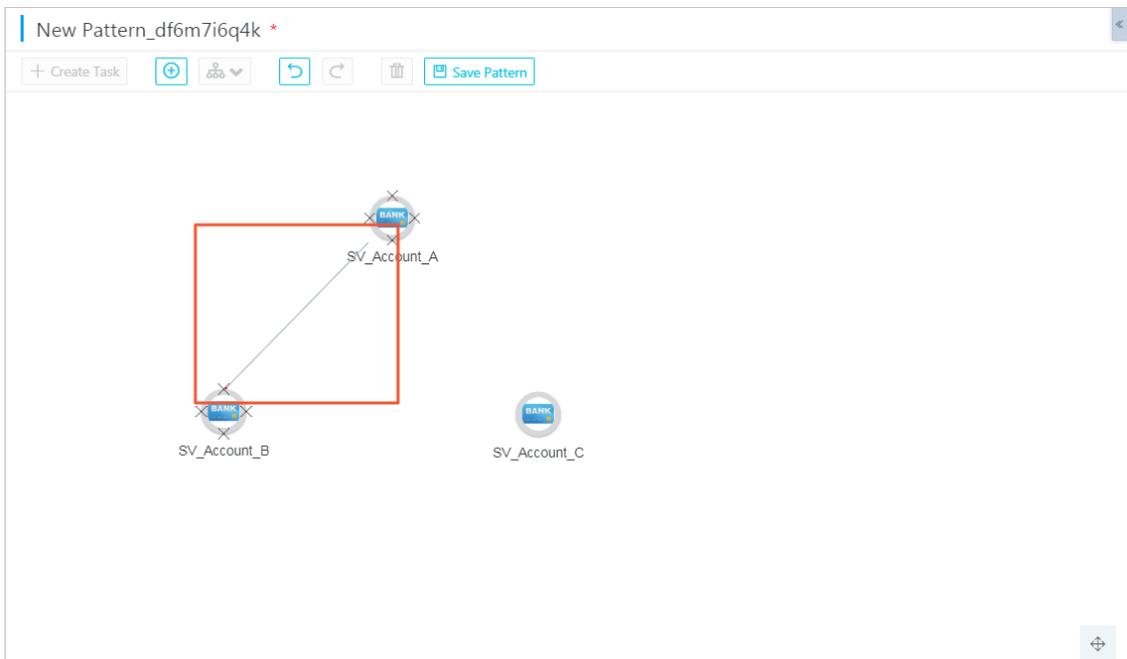


- iii. Click **OK**.
4. In the **Pattern Library** navigation pane, click the  icon (**Create Pattern**) or **Create Pattern** in the blank area on the right side.
5. Configure the relationship graph structure model of a pattern.

If an incorrect operation occurs, you can click the  icon (Undo) and the  icon (Redo) to quickly restore to a specific state.

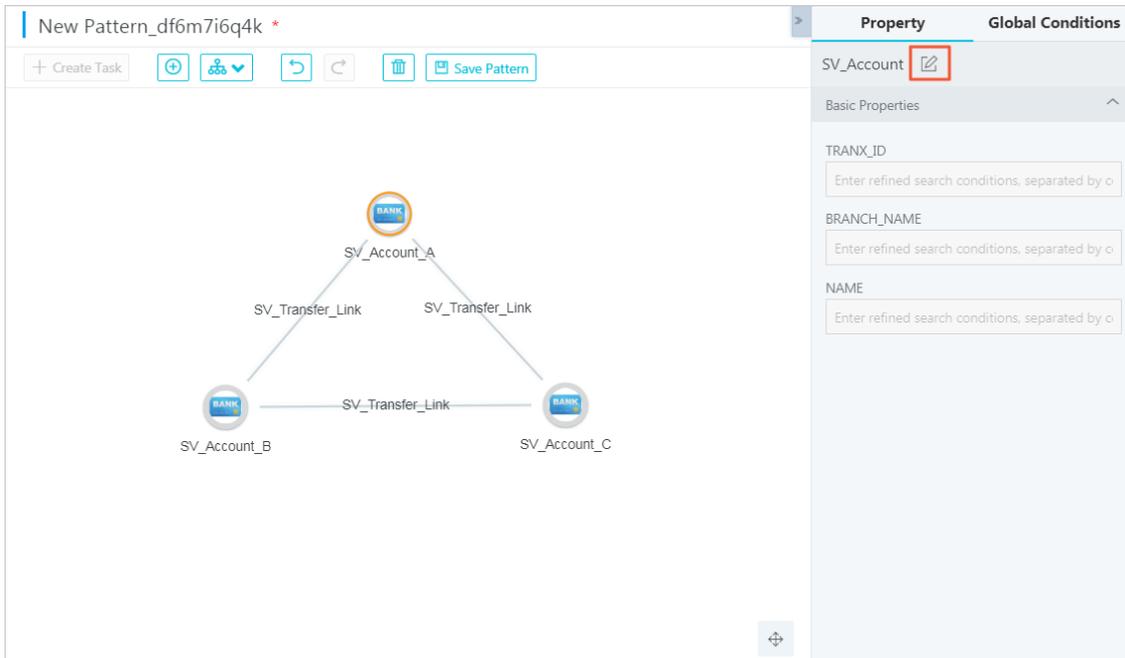
- i. For the new pattern **New pattern_xxx**, click the  icon in the toolbar, or click **Toolbox** in the left-side navigation pane to open the tool box.
 - ii. Drag the objects one by one from the toolbox to the right side of the page to add object nodes to **New mode_xxx**.
 - iii. Right-click the object node and select **Change Node Name** to set the name of the object node.

- iv. Move your mouse pointer to the source object. When four cross signs appear around the object, move your mouse pointer to a cross sign and press the left mouse button. Then, drag the mouse pointer to the target object. When four cross signs appear around the target object, release the left mouse button.

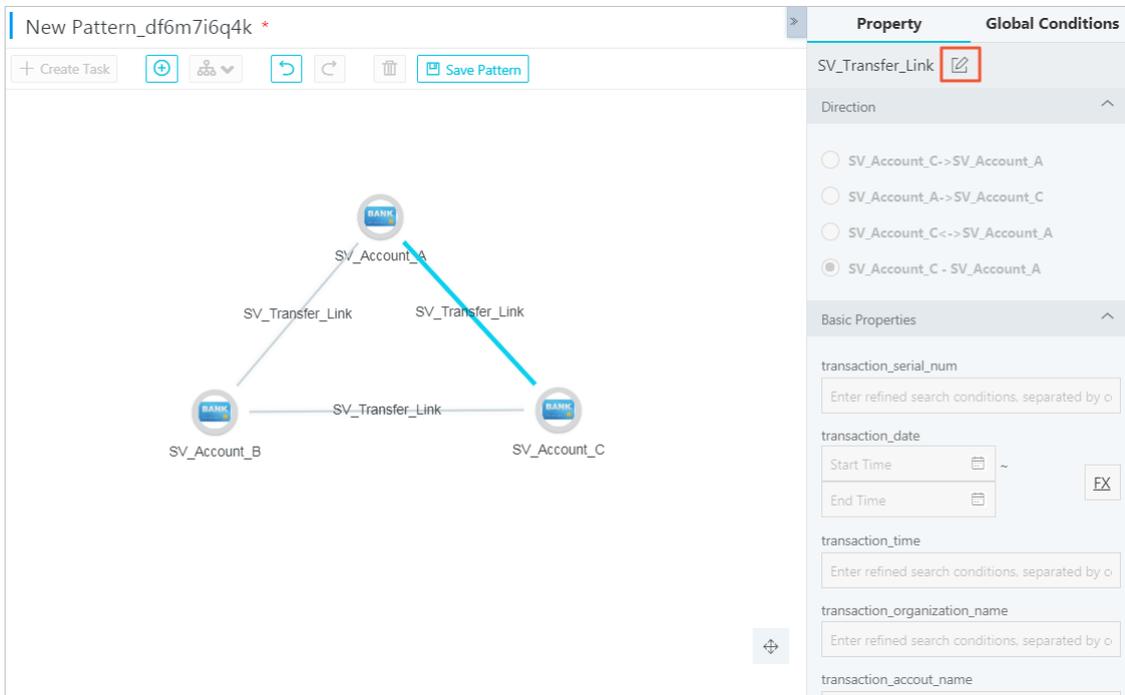


- v. In the **Add Link** dialog box that appears, select the link type and specify the link logic.
 - vi. Click **OK**.
 - vii. (Optional) To handle multiple object nodes with complex relationships, you can select the corresponding object node and click the  icon to select an appropriate layout.
6. Configure the filter conditions in the **Property** tab.
- i. Click the  icon in the upper-right corner, and then click **Property**.

- ii. Select an object. On the **Property** tab, click the  icon to set the filter conditions for the object.



- iii. After you have configured the preceding parameters, click the  icon to save the **Property** configurations of the object.
- iv. Select a link. On the **Property** tab, click the  icon to set the filter conditions and direction of the link.

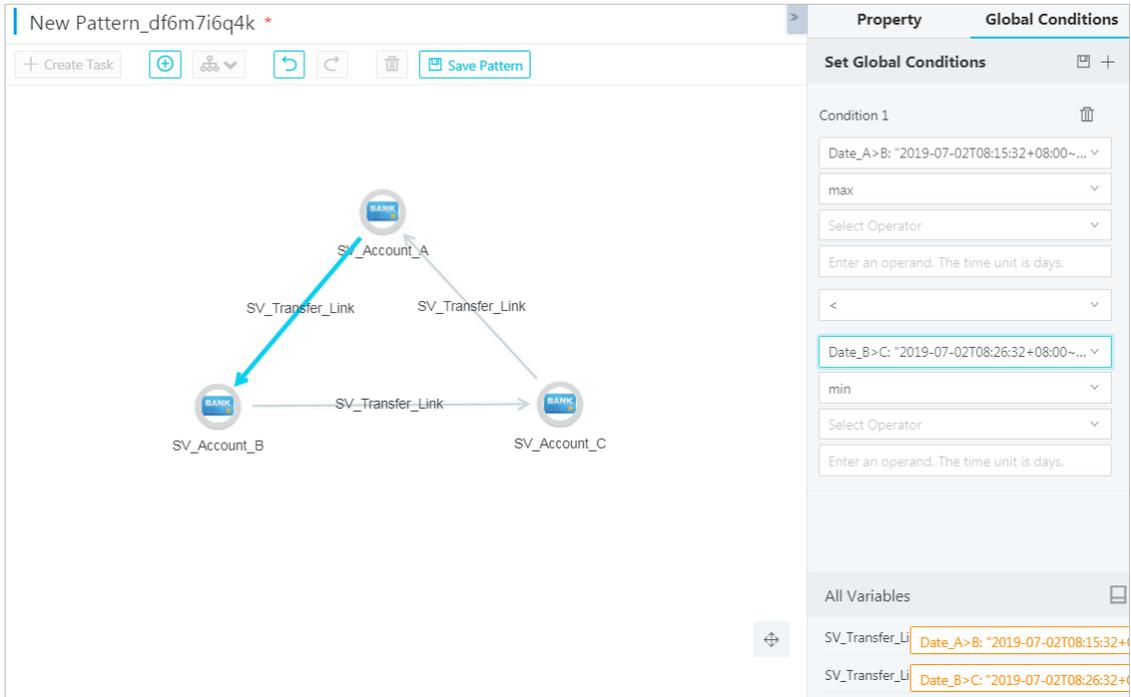


On the **Property** page of a link, properties marked with **FX** are the basis of **Global Conditions**.

- v. To set the global conditions based on a property marked with **FX**, click **FX** to name the property.

In this example: the **Transaction Date** property of the link between A and B is named as **Date A>B**. The **Transaction Date** property of the link between B and C is named as **Date B>C**.

- vi. After you have configured the preceding parameters, click the  icon to save the **Property** configurations of the link.
7. Configure filter conditions on the **Global Conditions** tab.
- i. Click the  icon in the upper-right corner, and then click **Global Conditions**.
 - ii. If you have not set the global conditions, click [Click here to configure settings](#).



On the **Global Conditions** tab, configure the global conditions based on the link properties that are marked with FX in the **Property** tab. The global condition in this example is that the transfer from A to B is made earlier than the transfer from B to C.

- iii. After you have configured the preceding parameters, click the  icon to save the configurations on the **Global Conditions** tab.
8. Click **Save Pattern**. Set the parameters in the **Create Pattern** dialog box that appears.
- If you cannot find a proper folder in **Target Directory**, you can click the  icon to create a folder.

9. Click OK.

8.14.2.2. View patterns

In Graph Analytics, you can view all the patterns within your permissions and details of each pattern, such as the relationship graph structures and tasks created based on the pattern.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Note Only administrators and creators can view the private patterns.

- You have created a pattern. For more information about how to create a pattern, see [Create patterns](#).

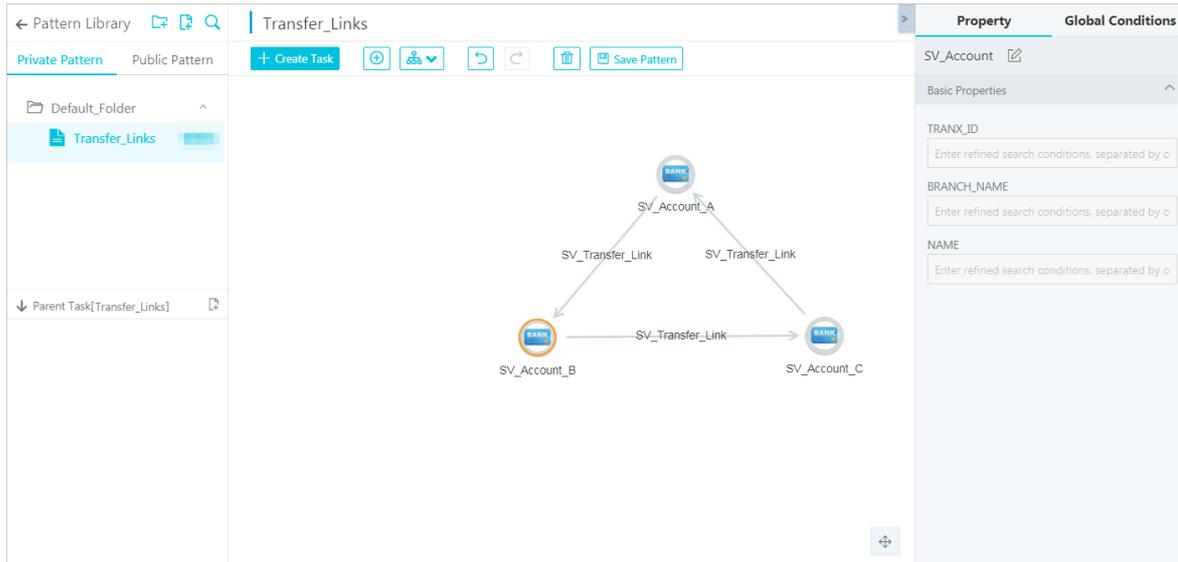
Context

You can view the following pattern information:

- Relationship graph structures of a pattern
- Tasks created based on a pattern
- Filter conditions of a pattern

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, click **Private Pattern**, and then click a folder to view the private patterns in the folder. Click **Public Pattern**, and then click a folder to view the private patterns in the folder.
4. Click a pattern to view its details.



Operation	Procedure
View the relationship graph structures	Click a pattern to view the relationship graph structures on the right side of the page.
View tasks created based on a pattern	Click a pattern. All tasks created based on this pattern are displayed under the Pattern Library tab.
View the filter conditions of a pattern	In the relationship graph structure, select an object or link, and click the  icon in the upper-right corner to display the Property tab and the Global Conditions tab. You can view the filter conditions of objects or links and the global filter conditions of the pattern.

8.14.2.3. Modify patterns

You can modify a pattern when its relationship graph structure and filter conditions are not suitable for the business scenarios. Modifying a pattern does not affect the tasks that are created based on the pattern.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can modify private patterns.

- You have created a pattern. For more information about how to create a pattern, see [Create patterns](#).

Context

In Graph Analytics, you can modify the following aspects of a pattern:

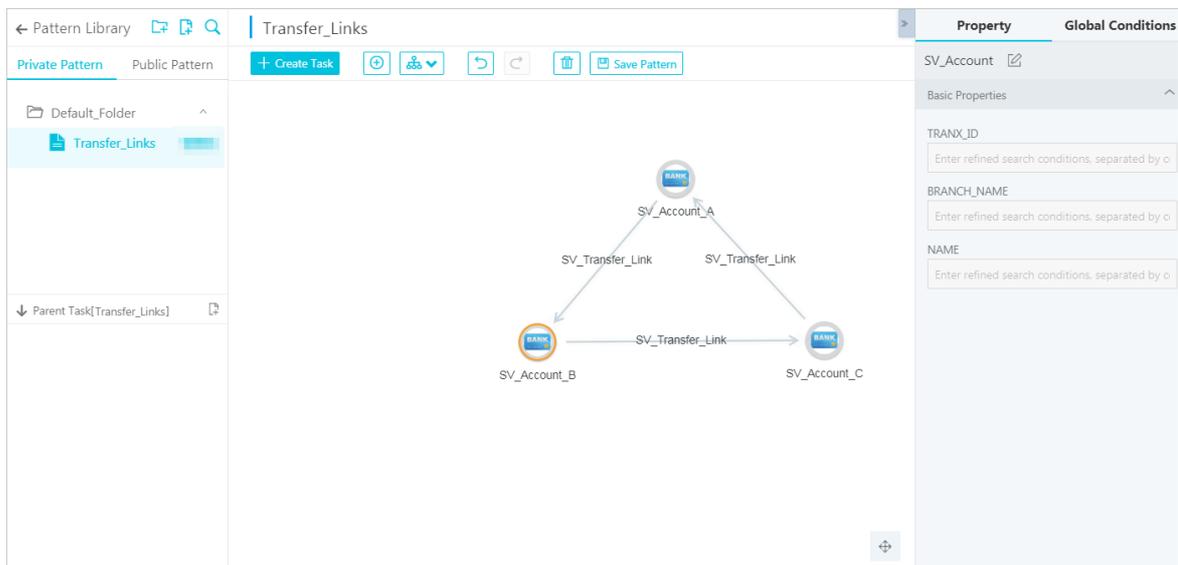
- Modify the pattern name
- Modify the filter conditions
- Modify the object node name
- Modify the link type
- Add an object node

- Add a link
- Delete an object node
- Delete links

Note After you modify a pattern, the existing tasks that are created based on this pattern are not affected. The modifications will be applied when you create a new task based on the modified pattern.

Procedure

1. Log on to Analytics Workbench.
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern**.



4. Modify the pattern as needed.

Operation	Procedure
Modify the pattern name	<ol style="list-style-type: none"> In the Pattern Library navigation pane, click Private Pattern or Public Pattern and select a pattern to be modified. Right-click the pattern and select Rename. Reset the name, and then press Enter. A message is displayed, indicating that the operation is successful.
Modify the filter conditions	Click the  icon in the upper-right corner, and set the filter conditions in Property and Global Conditions . For more information, see Create patterns .
Modify the object node name	<ol style="list-style-type: none"> Right-click the object node to be modified, and select Change Node Name. In the dialog box that appears, set the new name of the object node, and then click OK.

Operation	Procedure
Modify the link type	<ol style="list-style-type: none"> i. Right-click the link to be modified, and select Set Link Type. ii. In the dialog box that appears, select the link type, specify the link logic, and then click OK. You can select multiple link types.
Add an object node	<ol style="list-style-type: none"> i. Click the  icon in the toolbar, or click Toolbox on the left side of the page. ii. Drag the objects one by one from the toolbox to the right side of the page to add object nodes. iii. Add a link to a new object node.
Add a link	<p>After you have deleted a link or added a new object node, you need to add a link for the object node.</p> <ol style="list-style-type: none"> i. Move your mouse pointer to the source object. When four cross signs appear around the object, move your mouse pointer to a cross sign and press the left mouse button. Then, drag the mouse pointer to the target object. When the four cross signs appear around the target object, release the left mouse button. ii. In the Add Link dialog box that appears, select the link type and specify the link logic. iii. Click OK.
Delete object nodes or links	<div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p> Note After an object node is deleted, the links related to the object node are also deleted.</p> </div> <p>Method one:</p> <ol style="list-style-type: none"> i. Select an object node or link, and click the  icon in the toolbar. Or, you can press the Delete key. You can click-and-drag the mouse to box select multiple object nodes. ii. In the message box that appears, click Delete Selected. <p>Method two:</p> <p>Right-click an object node or link, or any of the object nodes in the box selection, and then select Delete Selected to directly delete the object node or link.</p>

5. After you have configured the preceding parameters, click **Save Pattern** in the toolbar to save the pattern.

8.14.2.4. Set private patterns to public patterns

In Graph Analytics, you can set private patterns to public patterns, but this is an irreversible operation.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can set private patterns to public patterns.

- You have created a private pattern. For more information about how to create a pattern, see [Create patterns](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, click **Private Pattern**, right-click a private pattern, and then select **Set as Public Pattern**.
4. In the dialog box that appears, configure relevant parameters.

The parameters are described in [Parameter configurations for setting a private pattern to a public pattern](#).

Parameter configurations for setting a private pattern to a public pattern

Parameter	Description
Publish the dependent task?	<p>Valid values are as follows:</p> <ul style="list-style-type: none"> ◦ Yes: Private tasks created based on this private pattern are also set to public tasks. ◦ No: Private tasks created based on this private pattern remain private tasks. After this private pattern is set to a public pattern, these tasks will be converted to private tasks that are created based on a public pattern. You can manually set these tasks as public tasks when necessary.
Target Directory	<p>Set the target directory in Public Pattern to receive the private pattern.</p> <p>When the private pattern is set as a public pattern, it will be moved from the directory in Private Pattern to the specified directory in Public Pattern.</p>

5. After you have configured the preceding parameters, click **OK**. A message is displayed, indicating that the operation is successful.

8.14.2.5. Delete a pattern

You can delete a pattern that is no longer used. Deleting a pattern does not affect the tasks that are created based on the pattern.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can delete private patterns.

- You have created a pattern. For more information about how to create a pattern, see [Create patterns](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, click **Private Pattern** or **Public Pattern** and select a pattern to be deleted. Right-click the pattern and select **Delete**.

4. In the message box that appears, click **OK**.

8.14.3. Tasks

8.14.3.1. Create a task

A task is created based on a pattern. It can be used to query the data with the same graph structure as the task in the data source.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.
- You have created a pattern. For more information about how to create a pattern, see [Create patterns](#).

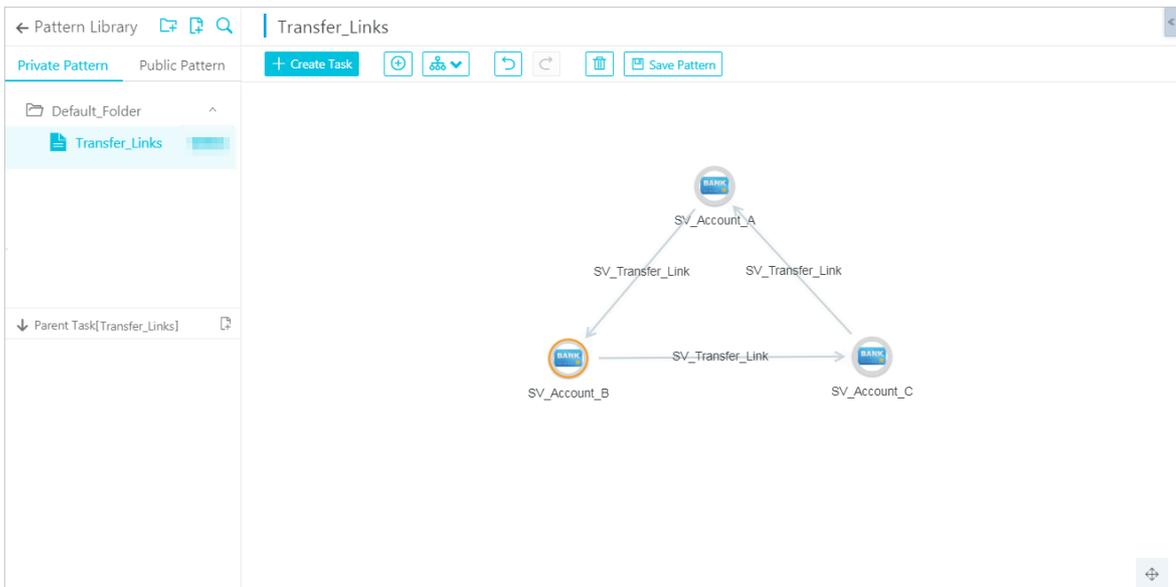
Context

For a task created based on the pattern, the objects, links, and filter conditions of the task are inherited from the pattern by default, which are exactly the same as the pattern. To modify these configurations, see [Modify a task](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern**. Click **Create Task** on the right side of the page or click the  icon (**Create a Task**).

 **Note** You can only create private tasks based on a private pattern. You can create public tasks or private tasks based on a public pattern.



The screenshot shows the 'Pattern Library' interface. On the left, there is a navigation pane with 'Default_Folder' and 'Transfer_Links'. The main area displays a graph pattern with three nodes: 'SV_Account_A' at the top, 'SV_Account_B' at the bottom left, and 'SV_Account_C' at the bottom right. All nodes are labeled 'BANK'. There are three edges, all labeled 'SV_Transfer_Link', connecting 'SV_Account_A' to 'SV_Account_B', 'SV_Account_A' to 'SV_Account_C', and 'SV_Account_B' to 'SV_Account_C'. The interface includes a top navigation bar with 'Pattern Library', search, and 'Transfer_Links' tabs. Below the tabs are buttons for '+ Create Task', '+ Save Pattern', and other icons. A 'Parent Task[Transfer_Links]' section is visible at the bottom left of the main area.

4. Configure the parameters on the **Create Task** dialog box that appears.
The parameters are described in [Parameter configurations for creating a task](#).

Parameter configurations for creating a task

Parameter	Description
Task Name	The user-defined task name.
Affiliated Pattern	The pattern that you have selected. It is automatically entered and cannot be modified.
Task Type	When the selected pattern is public, you can set the task type to Public Task or Private Task . When the selected pattern is private, the default task type is Private Task and cannot be modified.

5. Click **OK**.

After you have configured the preceding parameters, a message appears, indicating that the task has been created. The objects, links, and filter conditions of the task are inherited from the pattern by default, which are exactly the same as the pattern.

8.14.3.2. Check the task

You can view tasks based on patterns. You can select a specific pattern and view all the tasks that are created based on the pattern. You can also view the detailed information about these tasks, such as the relationship graph structures and filter conditions of the tasks.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can view the private tasks.

- You have created a task. For more information about how to create a task, see [Create a task](#).

Context

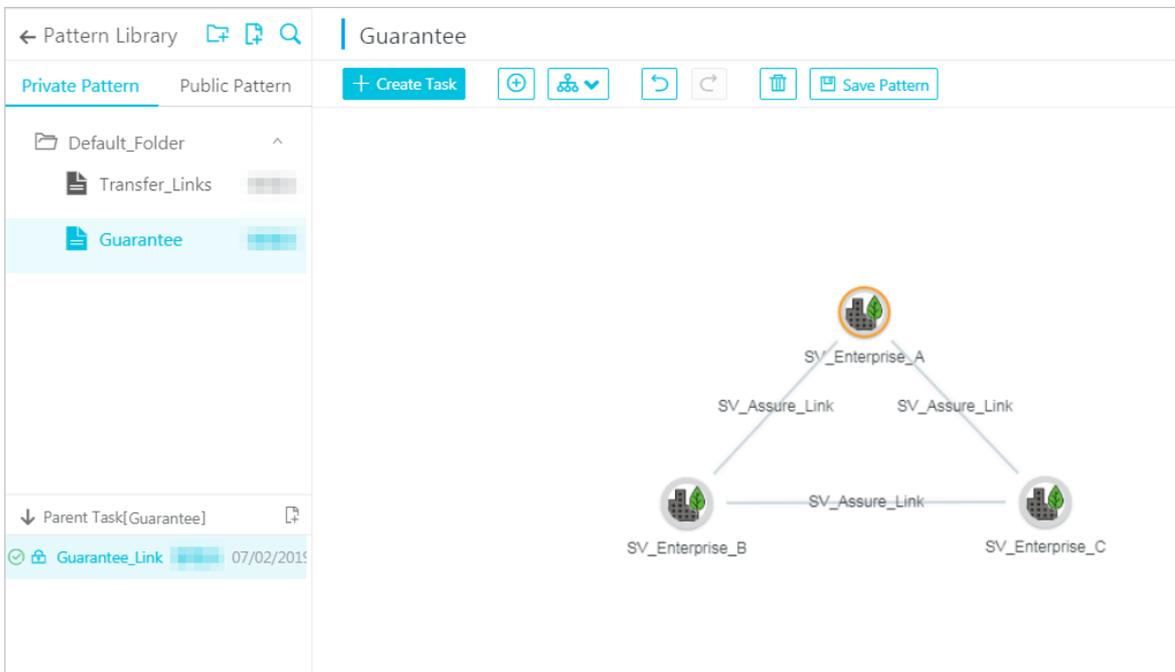
Tasks are divided into public tasks and private tasks. A lock icon is displayed before a private task, but no lock icon is displayed before a public task. A task can have three statuses: executed, not executed, and failed. These three statuses are represented by different icons.

You can view the following task information:

- The task type and task status
- Relationship graph structures of a task
- Filter conditions of a task
- Primary keys of the object in the task

Procedure

1. [Log on to Analytics Workbench.](#)
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern** and view the tasks that are created based on the pattern.



The task types and statuses are described as follows:

- Tasks are divided into public tasks and private tasks. A lock icon is displayed before a private task, but no lock icon is displayed before a public task.
 - A task can have three statuses: executed, not executed, and failed. These three statuses are represented by different icons.
4. Click a task and view the detailed information about the task as follows.

Operation	Procedure
View the relationship graph structures of a task	Click a specific task to view the relationship graph structures on the right side of the page.
View the filter conditions of a task	Select an object or a link, and click the  icon in the upper-right corner. The Property and Global Conditions tabs are displayed. You can view the filter conditions of an object or a link and the global conditions of tasks.
View the primary keys of objects in a task	Right-click an object and select Set Primary Keys to view the primary keys that have been set for this object.

8.14.3.3. Modify a task

For a task that is created based on a pattern, the objects, links, and filtering conditions of the task are inherited from the pattern by default, which are exactly the same as the pattern. When some configurations of a task do not match the scenario, you can modify the task accordingly.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can modify private tasks.

- You have created a task. For more information about how to create a task, see [Create a task](#).

Context

In Graph Analytics, you can modify the following aspects of a task:

- Modify the filter conditions
- Modify the object node name
- Modify the link type
- Add an object node
- Add a link
- Delete an object node
- Delete a link

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern**. Select the task to be modified.
4. Modify a task as shown in the following operations.

Operation	Procedure
Modify the filter conditions	You can click the  icon in the upper-right corner. You can set the filter conditions for a task in the Property and Global Conditions tabs that appear. For more information, see Create patterns .
Set the object primary key	<ol style="list-style-type: none"> i. Right-click the object node to be modified, and select Set Primary Keys. ii. In the dialog box that appears, set the primary keys of the object node, and click OK. You can set multiple primary keys.
Modify the object node name	<ol style="list-style-type: none"> i. Right-click the object node to be modified, and select Change Node Name. ii. In the dialog box that appears, set the new name of the object node, and then click OK.
Modify the link type	<ol style="list-style-type: none"> i. Right-click the link to be modified, and select Set Link Type. ii. In the dialog box that appears, select the link type, specify the link logic, and then click OK. You can select multiple link types.
Add an object node	<ol style="list-style-type: none"> i. Click the  icon in the toolbar, or click Toolbox on the left side of the page. ii. Drag the objects one by one from the toolbox to the right side of the page to add object nodes. iii. Add a link to a new object node.

Operation	Procedure
Add a link	<p>After you have deleted a link or added a new object node, you need to add a link for the object node.</p> <ol style="list-style-type: none"> i. Move your mouse pointer to the source object. When four cross signs appear around the object, move your mouse pointer to a cross sign and press the left mouse button. Then, drag the mouse pointer to the target object. When the four cross signs appear around the target object, release the left mouse button. ii. In the Add Link dialog box that appears, select the link type and specify the link logic. iii. Click OK.
Delete object nodes or links	<div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #add8e6;"> <p> Note After an object node is deleted, the links related to the object node are also deleted.</p> </div> <p>Method one:</p> <ol style="list-style-type: none"> i. Select an object node or link, and click the  icon in the toolbar. Or, you can press the Delete key. You can click-and-drag the mouse to box select multiple object nodes. ii. In the message box that appears, click Delete Selected. <p>Method two:</p> <p>Right-click an object node or link, or any of the object nodes in the box selection, and then select Delete Selected to directly delete the object node or link.</p>

5. Click **Run Global Task** to save the modification configuration.

 **Note** After a task is modified, you must execute the task again. Otherwise, the modifications cannot be saved.

8.14.3.4. Execute the task and view the result

After the task is created or modified, you can execute the task to query sub-graphs that have the same graph structure as the task. The task execution results are displayed in a list. You can also view the execution results in the graph area.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can execute private patterns.

- You have created a task. For more information about how to create or modify a task, see [Create a task](#) and [Modify a task](#).

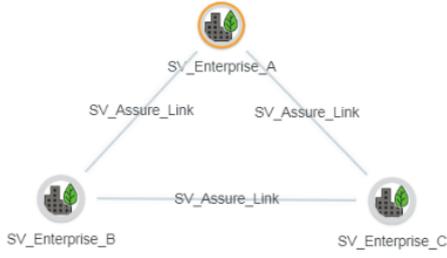
Procedure

1. [Log on to Analytics Workbench](#).

- In the top navigation bar, click **Intelligent Network**.
- In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern**. Select the task to be executed.
- Click **Run Global Task** in the top toolbar.
- In the right-side area, click the  icon to view the execution results.

Guarantee_Link

Run Global Task     



Result

Task Status: **Success** Operator: admin Duration: 1 s Matching Results: 4 Start Time: 07/02/2019 15:16:35 End Time: 07/02/2019 15:16:36

Guarantee_Link152_0

<input type="checkbox"/>	SV_Enterprise_A	SV_Enterprise_B	SV_Enterprise_C	Actions
<input type="checkbox"/>	27002	27003	27004	Go to Graph Go to Map
<input type="checkbox"/>	27003	27004	27008	Go to Graph Go to Map
<input type="checkbox"/>	27005	27006	27007	Go to Graph Go to Map

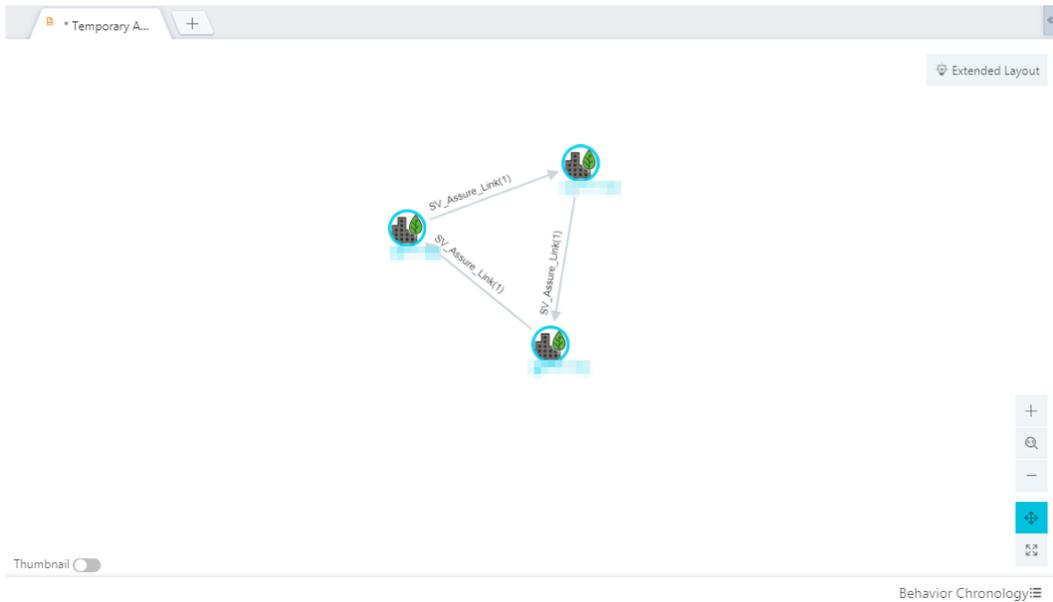
[Go to Graph](#) [Go to Map](#) Selected 0/4 1

- View the task execution results in the graph area.

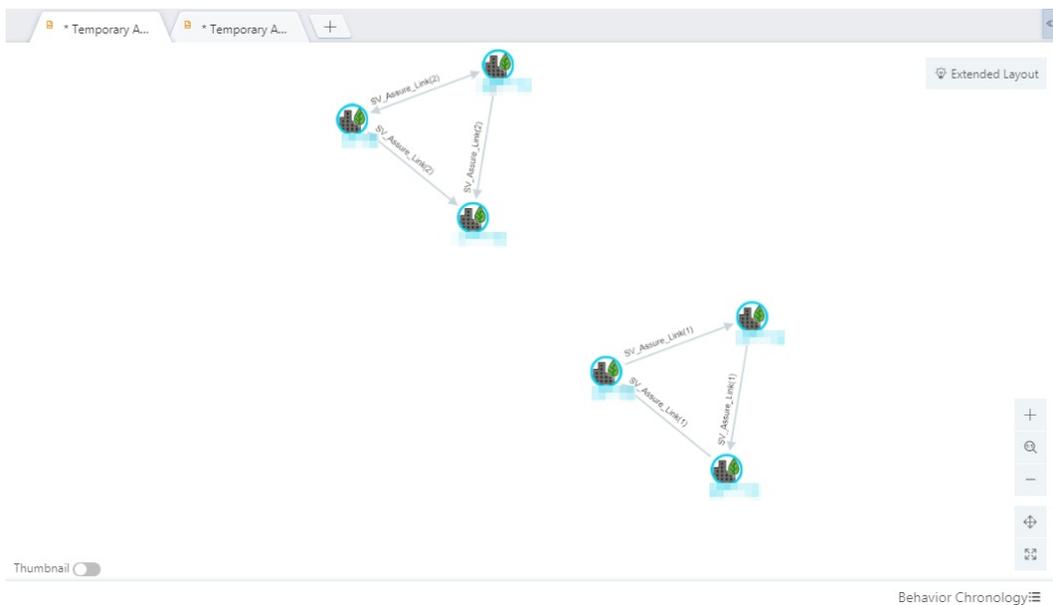
 **Note** The task result is displayed as a temporary analysis file in the graph area. You can save and share the file and perform other operations. You can also extend the analysis.

Operation	Procedure
View the results of a single task in the graph area	Select a record and click Go to Graph . The result is displayed on the graph page, as shown in View the results of a single task in the graph area .
View the results of multiple tasks in the graph area	You can also select multiple records and then click Go to Graph . The results are displayed on the graph page, as shown in View the results of multiple tasks in the graph area .

View the results of a single task in the graph area



View the results of multiple tasks in the graph area



8.14.3.5. Set a private task as a public task

Private tasks created based on public patterns can be set to public tasks, but this is an irreversible operation.

Prerequisites

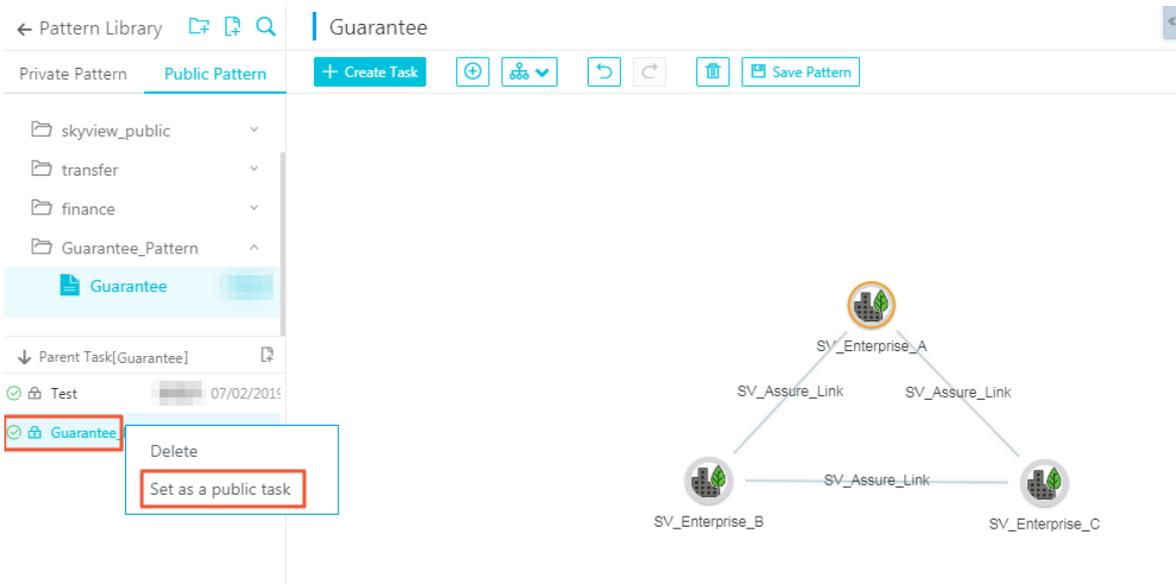
- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

Note Only administrators and creators can set private patterns to public patterns.

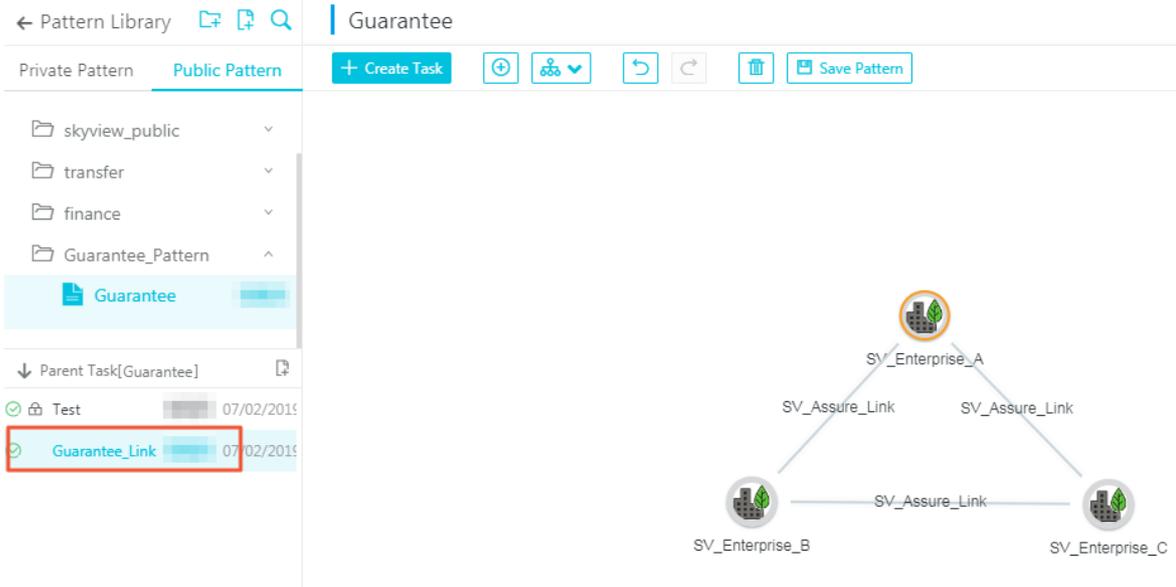
- You have created a private task based on the public pattern. For more information about how to create a pattern, see [Create a task](#).

Procedure

1. Log on to Analytics Workbench.
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, click **Public Pattern** . Right-click a private task that is created based on the pattern, and then select **Set as Public Task**.



After a private task is set to a public task, the lock icon in front of the task disappears. A message is displayed, indicating that the operation is successful.



8.14.3.6. Delete a task

You can delete a task when it is no longer needed.

Prerequisites

- Make sure that you have an account and password for Graph Analytics and are authorized to operate on Analytics Workbench and Administration Console.

 **Note** Only administrators and creators can delete private patterns.

- You have created a task. For more information about how to create a pattern, see [Create a task](#).

Procedure

1. [Log on to Analytics Workbench](#).
2. In the top navigation bar, click **Intelligent Network**.
3. In the **Pattern Library** navigation pane, select a pattern in **Private Pattern** or **Public Pattern**. Right-click the task to be deleted and then click **Delete**.

 **Note** Deleted tasks cannot be restored. Exercise caution when you delete a task.

4. In the dialog box that appears, click **OK**.

8.15. Examples

8.15.1. Tax industry case studies

Tax inspectors used Graph Analytics to analyze abnormal information and uncovered a tax cheat group that was related to 13 companies.

Case background

On April 1, 2016, a clerk of an overseas trading platform was routinely checking goods being loaded and unloaded. When the clerk checked "Hangzhou Children's Products Co., Ltd.", he was told that the goods had been shipped. The company exported goods such as children's products to overseas countries with a total volume of 40 million RMB. Apparently, this was a case of avoiding "loading inspection", and this case attracted the clerk's attention. The clerk used Graph Analytics to analyze the company and found a large-scale tax cheat group related to this company.

Data preparation

 **Note** All the data in this case, including individual people's names, company names, places, and times, is purely fictitious.

Data resources used in this case include:

- Information of the drawer (the manufacturer), the customer (the domestic trader), the remitter, and the overseas trader.
- The relationships between the customer and the drawer, the customer and the remitter, and the customer and the overseas trader are required.

Connect to data sources and configure an OLEP model

Before you perform an analysis, you must connect the data source to Graph Analytics and build an OLEP model based on the data table in the data source.

1. [Log on to Administration Console of Graph Analytics](#). For more information about how to connect the data source to Graph Analytics, see [Create data sources](#).
2. Create objects based on the data table in this case, as shown in [Create OLEP models for tables](#).

The created objects are as follows.

Objects mapped by the data table

Data table	Object
Drawer information table	<ul style="list-style-type: none"> Drawer: Maps the drawer information. WIFI: Maps the WIFI used by the drawer.
Customer information table	<ul style="list-style-type: none"> Contact phone number: Maps the telephone number of the customer. Customer: Maps the customer information.
Remitter information table	Remitter: Maps the remitter information.
Overseas trader information table	<ul style="list-style-type: none"> Country: Maps the country where the overseas trader is located. Overseas trader: Maps the overseas trader information.

3. Create a first-degree link mapping for the data table in this case, as shown in [Create OLEP models for tables](#).

The first-degree link is created as follows.

Links mapped by the data table

Data table	First-degree link	Source object	Target object
Drawer information table	<ul style="list-style-type: none"> Frequent logon WIFI: The link between the drawer and the frequent logon WIFI can be used to query both the drawer and the WIFI information. Name of the drawer's company: The link between the drawer and the drawer's company can be used to query both the drawer and the company name. Drawer's phone number: The link between the drawer and the drawer's phone number can be used to query both the drawer and the phone number of the drawer. 	<ul style="list-style-type: none"> Frequent logon WIFI: The drawer Name of the drawer's company: The drawer 	<ul style="list-style-type: none"> Frequent logon WIFI: The WIFI Name of the drawer's company: The name of the company

Data table	First-degree link	Source object	Target object
Customer information table	<ul style="list-style-type: none"> Customer's phone number: The link between the customer and the customer's phone number can be used to query both the telephone number or the user who have used this phone number. Name of the customer's company: The link between the customer and the customer's company can be used to query both the customer or the company of the customer. 	<ul style="list-style-type: none"> Customer's phone number: The customer Name of the customer's company: The customer 	<ul style="list-style-type: none"> Customer's phone number: The phone number Name of the customer's company: The name of the company
Customer-drawer information table	Invoicing: This link can be used to query both the customer and the drawer.	Drawer	Customer
Customer-remitter information table	Remittance: This link can be used to query both the customer and the remitter.	Remitter	Customer
Customer-overseas trader information table	Purchase: This link can be used to query both the overseas trader and the customer.	Overseas trader	Customer
Overseas trader information table	Country: This link can be used to query both the overseas trader and the country where the overseas trader is located.	Overseas trader	Country

4. For more information about how to configure the business information related to these objects and links, see [Configure object properties and business parameters](#) and [Configure link properties and business parameters](#).

5. Create multi-degree links as shown in [Create a multi-degree link](#).

The multi-degree links created in this case are as follows.

Multi-degree links

Multi-degree link	Base links	Description
The drawer and the customer with the same name.	<ul style="list-style-type: none"> Name of the customer's company. Name of the drawer's company. 	Queries the drawer with the same name as the customer.

Analysis process

After you have connected the data source and configured the OLEP model, you can go to Analytics Workbench to analyze cases.

1. [Log on to Analytics Workbench.](#)

2. On the **Graph** page, create a new analysis, and add **Hangzhou Children's Products Co., Ltd** as the node to start with.

3. Select the added node, and query the company's information on the right side of the page.

Results: This company is a manufacturer company instead of a trading company.

Inferences made by analysts: This company needs a domestic trader to export its goods.

4. Analysts used the link extension feature of Graph Analytics to query the downstream customer of this company, namely the domestic trader.

Select this company as a node, and click **Link Extension** in the toolbar. Set **Link Type** to **Invoicing** to query the downstream customer of this company.

Results: This company has been the drawer for three trading companies. These three companies are from the medical industry and the apparel industry, but somehow have become the drawee of a manufacturer engaged in children's products.

5. Analysts used the link extension feature to investigate these three customers to find the overseas traders that have built business relationships with these three customers.

Select these three customers, and click **Link Extension**. Set **Link Type** to **Purchase** to query the overseas traders that have built business relationships with these three customers.

Results: All these customers have their own overseas trader.

6. Analysts used the link extension feature to search the remitter of these three customers.

Select these three customers, and click **Link Extension**. Set **Link Type** to **Remitter**, and search for the remitters related to these three customers.

Results: The analysis results indicate that these three customers receive remittance from the same remitter. In this phase, the relationship network is downward-trend and looks like a symmetrical funnel, which is a typical abnormal pattern.

Inferences made by analysts: A manufacturer sells products to overseas buyers through three customers, but there is only one remitter. Whether these three overseas traders are in the same area or not, this network pattern is very suspicious.

7. Analysts tried to query the locations of these three overseas traders.

Select these overseas traders, and click **Link Extension**. Set **Link Type** to **Country** to query the locations of these three overseas traders.

Results: These three overseas traders are located in different countries.

Inferences made by analysts: These overseas traders are located in different countries, but the remittance is made by the same remitter. Given that, it is very likely that these three customers and the drawer have committed tax cheating.

8. To obtain more information, analysts viewed the information cubes of these three customers.

Results: Two of the customers are companies that provide both manufacturing and trading services.

9. Following these clues, analysts queried the manufacturers of these two companies.

Select these two companies and click **Link Extension**. Set **Link Type** to **Drawer and customer with the same name** to query the manufacturers of these two companies.

Results: These two companies share the same manufacturer (drawer).

10. Analysts used **Group Analysis** to analyze the relationship between these two companies.

Select these two nodes and their manufacturers, and choose **Analyze > Group Analysis**. Set **Link Type** to **Invoicing** to analyze the relationship between them.

Results: These two companies issue invoices to each other.

Judgement made by the analysts: Based on experience, this phenomenon is obviously abnormal.

11. Analysts used **Backbone Analysis** to verify whether this network contains any key members.

In the toolbar, choose **Analyze > Backbone Analysis**. Set the **Backbone Node** to **Customer** to analyze the backbone customers in the current network.

Results: The current relationship network contains two companies as key nodes. These two companies have taken up important positions in the current relationship network.

12. Analysts used the link extension feature to search for the drawers of these two key members.

Select these two key members, and click **Link Extension**. Set **Link Type** to **Invoicing** to query the drawers of these two key members.

Results: Analysts found a group of drawers by analyzing one of the key members.

Inferences made by analysts: This company provides both manufacturing and trading services. It provides trading services for domestic manufacturers from various industries, including the electromechanics industry, electronics industry, moulding industry, and the food industry. This is a suspicious issue of tax cheating.

13. Analysts checked the behavior details of these drawers and the key member.

Select this key member and the group of drawers. Click **Behavior Chronology**, and then click the **Details** tab. Set **Link** to **Invoicing** to view the invoicing details.

Results: The proportion of sales and export volumes are both around 10%, which is relatively even.

Inferences made by analysts: The proportion of orders and exports are too even, which shows obvious human manipulation.

14. Analysts used **Common Neighbors** to check whether these drawers have shared objects, for example, WIFI.

Select the drawer nodes, and choose **Analyze > Common Neighbors**. Set **Link Type** to **WIFI** to check whether the drawers have a shared WIFI.

Results: These drawers often connect to WIFI using the same MAC address to log on to the trading platform.

Inferences made by analysts: This is an obvious abnormal phenomenon. It is very likely that these companies are operated by the same group of people.

15. Analysts used **Common Neighbors** to check whether the drawers and the customers have shared phone numbers.

Select all drawers and customer nodes, and choose **Analyze > Common Neighbors**. Set **Link Type** to **Drawer TEL** and **Customer TEL** to check whether the drawer and the customer have a shared contact number.

Results: A telephone number was shared by different drawers and customers.

Inferences made by analysts: Based on the abnormal information found in the relationship network, it is very likely that these groups are cheating on tax rebates.

On-the-spot investigation

According to the results returned from the on-the-spot investigation, all these manufacturers and traders are highly suspicious. The department concerned continued to investigate these groups, and uncovered a tax cheat group related with 13 enterprises. This group registered different roles on the trading platform, made fake invoices and transactions internally, and the amount of tax refunds reported to the trade platform reached 100 million RMB.

8.16. FAQ

1. Q: How can I log on to the system for the first time?

A: Contact the administrator to obtain the account and the initial password to log on to Graph Analytics. To keep your account secure, modify the password as prompted.

2. Q: Why does the system prompt an error message indicating incorrect user name and incorrect password when I was logging on to Graph Analytics?

A: If you are unable to log on to the system, check whether the account name and the password you entered are correct, and whether the password is entered in half-width characters, in the correct case, or has any space. If the error persists, contact the administrator.

3. Q: What is the default rule for the simple link extension (double-clicking a link to start a link extension)?

A: To help you analyze the information quickly, the administrator configures common links in Administration Console and synchronizes them to the double-click operation. These common links must be published by the administrator. If you need to add or remove a double-click link extension, contact the administrator.

4. Q: Why does the system prompt "a maximum of five nodes can be selected" when I select all nodes in the graph area to perform the link lookup analysis?

A: By default, Graph Analytics supports a maximum of five nodes in the link extension analysis. If you need to analyze more than five nodes, contact the administrator, and the administrator will assess the system scale and make adjustments.

5. Q: Why can't I delete a node by pressing Delete on the keyboard?

A: Currently, you cannot delete a node by pressing Delete on the keyboard. You can right-click the selected node and click **Delete** in the shortcut menu.

6. Q: How many steps can be undone or rolled back?

A: A maximum of 20 steps can be rolled back or undone.

7. Q: Why does the Path Analysis button turn gray after I select a node?

A: Path Analysis is available only after two nodes are selected.

8. Q: After I select a node, why do the Group Analysis button and the Common Neighbors button turn gray?

A: Group Analysis and Common Neighbors are available only after two or more nodes are selected.

9. Q: Can I configure a new link analysis method discovered by myself while using Graph Analytics?

A: Currently, links are configured by the administrator. If you need to add a new link analysis method, contact the administrator.

10. Q: Why is there no data available after I click Behavior Analysis, Chronology Analysis, and Details?

Q: You need to select a node in the graph area before performing these analyses or viewing behavior details.

11. Q: Why are statistics displayed on the right side when no node in the graph area is selected?

A: If no node is selected, statistics on all nodes are collected by default.

12. Q: How to find the nodes that fall out of the scope of the canvas?

A: You can enable the thumbnail to locate the nodes, and move the nodes that fall out of the canvas to the visible area.

13. Q: Why are all layout buttons become gray and unavailable?

A: The layout buttons become available only after you select a node in the graph area.

14. Q: After I select all the nodes in the graph area, why does the system give no response after I click Hierarchical Layout?

A: In the Hierarchical Layout mode, you need to select one node as the starting point to view the hierarchical layout.

15. Q: Besides the drag button in the toolbar, can I use any short cut operations to move the canvas?

A: You can press **Space** on the keyboard to move the canvas.

9. Dataphin

9.1. What is Dataphin?

Dataphin is an intelligent engine for building big data platforms. It is designed to meet the requirements of big data development, management, and application across multiple industries. Dataphin combines technologies and methodologies. It provides all-in-one intelligent data development and management services, including data ingestion, data standardization, data modeling, data asset management, and data services.

Dataphin applies to different computing and storage environments. This enables you to use a single console to process data from various data sources. Dataphin allows you to import data, standardize data production, develop data by data modeling, and create a tag system by extracting tags from entities. You can also generate and manage data assets by using your business data and knowledge. Dataphin also provides multiple types of data services such as table query and intelligent voice search.

9.2. Usage notes

This topic describes the notes on using Dataphin.

To use Dataphin, you must have the necessary knowledge and expertise. This document is intended for:

- Application developers
- Analysts
- Data developers
- Technical architects
- System administrators

To ensure stable running of Dataphin, observe the following limits and recommendations.

Operation	Limit and recommendation
Set the computing engine type	Select a computing engine type and configure the cluster where your computing engine resides. For example, specify the endpoint of the cluster. MaxCompute and Hadoop are the available computing engine types. The system uses the selected computing engine type to support data construction in the specified cluster. Select the computing engine type and configure the computing engine settings based on your computing engine cluster.
Create a data source	<ul style="list-style-type: none">• We recommend that you configure an AccessKey pair with administrative privileges for data source management.• One physical database can be used only as one data source.
Create a project	<ul style="list-style-type: none">• When the data source type is MaxCompute, the project name must be the same as your MaxCompute project name.• The project name cannot start with <code>LD_</code> or <code>ld_</code>. If project names start with <code>LD_</code> or <code>ld_</code>, project names conflict with business unit names. In this case, errors may occur during table query because the system cannot differentiate between a logical table and a physical table. In Dataphin, when you query a logical table, you must prefix the table name with the corresponding business unit name. When you query a physical table, you must prefix the table name with the corresponding project name.

Operation	Limit and recommendation
Configure the computing engine	<ul style="list-style-type: none"> • If a physical database is configured as a computing engine for a project, we recommend that you do not add, delete, or modify data in the database in the consoles of other cloud services. • We recommend that you do not configure computing engines of the same type that reside in different clusters for a project.
Process data	Computing engines can read data only from their own clusters.
Perform data standardization and data modeling	<ul style="list-style-type: none"> • We recommend that you exercise caution when you name data standardization objects and logical tables. Set names in lowercase for data standardization objects and make sure that the names are valid and easy to read. The names cannot be changed when they have downstream dependencies. • Use abbreviations whenever possible to avoid data production errors. Errors may occur if the length of a field name exceeds the limit that is imposed by the database.
Run an ad hoc query	When you query a logical table, prefix the table name with the corresponding business unit name. When you query a physical table, prefix the table name with the corresponding project name.
Extract the data value	We recommend that you set IDs based on user information to ensure an accurate match between an ID and a user.

9.3. Quick start

9.3.1. Instructions for system administrators

To prepare Dataphin for use, system administrators and deployment engineers must make sure that the environment is ready and required user roles are created.

Procedure

1. Make sure that the following environment and resources are ready:
 - Apsara Infrastructure Management Framework is deployed. The Apsara Uni-manager Management Console is accessible.
 - MaxCompute, Object Storage Service (OSS), DAuth, Server Load Balancer (SLB), Elastic Compute Service (ECS), three physical machines, and PostgreSQL database resources are available.

 **Note** The PostgreSQL database in your environment must be accessible. This PostgreSQL database can be deployed on an ApsaraDB RDS for PostgreSQL instance or an ECS instance:

- ApsaraDB RDS for PostgreSQL: A 6U model is required for Apsara Stack V3.7.0 and earlier versions. For versions later than V3.7.0, a 7U model is required. We recommend that you do not use miniRDS because it will soon become obsolete.
- PostgreSQL on ECS: Dataphin provides this configuration in Apsara Stack V3.7.0 and later versions. You can obtain this baseline configuration if you purchase one of these versions. If you purchase a version earlier than V3.7.0, we recommend that you contact the Dataphin team to obtain specific ECS resources and deploy PostgreSQL in a non-standard way.

2. Obtain the following information about the computing engine cluster:

- MaxCompute endpoint: This information is required when you set the computing engine type.
- AccessKey ID and AccessKey secret that are used to create the Dataphin_Meta project in MaxCompute: This MaxCompute project is used to compute and store MaxCompute metadata.

 **Note**

After Dataphin and MaxCompute are deployed, a MaxCompute project is created to obtain MaxCompute metadata in the Apsara Stack environment. System administrators and deployment engineers must verify that the Dataphin_Meta project exists and obtain the AccessKey ID and AccessKey secret that are used to create this MaxCompute project. If the Dataphin_Meta project does not exist, contact deployment engineers to create one and grant Dataphin the permission to obtain MaxCompute metadata.

3. Create user roles. After Dataphin is deployed, the account system provides the following three types of user roles in the Apsara Stack environment:
 - O&M super administrator: an independent metadata management tenant of the system. The O&M super administrator can obtain and parse the metadata of your cluster. Each system has only one O&M super administrator. Contact deployment engineers to obtain the username and password of the O&M super administrator.

 **Note** System administrators must keep the username and password of the O&M super administrator confidential, and exercise caution after logon as the O&M super administrator.

- Super administrator: a tenant who performs construction operations. For example, the super administrator can manage users and also design and build the architecture of a specific business data system. In the Apsara Stack environment, the role of a department account or Apsara Stack tenant account is the super administrator.
- Regular user: a member of a department. A regular user performs development operations, including the detailed design of a specific business data system. In the Apsara Stack environment, a RAM user can be created within a department account. A RAM user can be added as a tenant member of the super administrator.

 **Note**

Each user can belong to only one tenant. A user from one department cannot be added as a tenant member of another department. We recommend that each Dataphin system be used only for one department, namely, one tenant.

9.3.2. Log on to the Dataphin console

This topic describes how to log on to the Dataphin console in the Apsara Uni-manager Management Console.

Prerequisites

- The IP address or domain name of the Apsara Uni-manager Management Console is obtained from deployment engineers before you log on to the Apsara Uni-manager Management Console. The URL that you can use to log on to the Apsara Uni-manager Management Console is in the following format: `https://IP address or domain name of the Apsara Uni-manager Management Console`.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.

2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username as prompted. For security concerns, your password must meet the minimum complexity requirements. The password must be 8 to 20 characters in length and must contain at least two types of the following characters: letters, digits, and special characters such as exclamation points (!), at signs (@), number signs (#), dollar signs (\$), and percent signs (%).

3. Click **Login** to go to the Apsara Uni-manager Management Console homepage.
4. In the top navigation bar, choose **Products > Big Data > Dataphin**.
5. Select the organization and region. Then, click **Dataphin**.

9.3.3. Preparations

This topic describes the preparations that you must complete before you perform the operations that are described in Quick start.

Prepare for using Dataphin

1. Specify the endpoint of the computing engine cluster for Dataphin. For more information, see [Set the computing engine type](#).
2. Create a project and configure the sandbox whitelist. For more information, see [Create projects](#).
3. Create a data domain named test_dataphin. For more information, see [Create a data domain](#).
4. Create a table named dataphin_test as the destination table of data synchronization. For more information, see [Create a destination table for data synchronization](#). You can execute the following SQL statement to create the destination table:

```
CREATE TABLE IF NOT EXISTS `datax_test` (order_id bigint comment 'Order ID',`area` string comment 'Region',province string comment 'Province',city string comment 'City',product_type string comment 'Type',order_name string comment 'Customer name',report_date datetime comment 'Date',order_amt double comment 'Sales amount')PARTITIONED BY ( `ds` STRING);
```

5. Add members to the project. For more information, see [Add members](#). In this tutorial, add two members and set their roles to **Developer** and **Project Administrator**.

Prepare a data source

In this tutorial, statistics are collected on the total daily sales of office supplies and technical products in each province.

1. Create an ApsaraDB RDS data source. For more information, see Quick start in *ApsaraDB RDS User Guide*. In this tutorial, the ApsaraDB RDS data source is named dataphin.
2. Download the [company_sales_record_copy](#) table to be used in this tutorial.
3. Upload the company_sales_record_copy table to the ApsaraDB RDS data source that you have created.

9.3.4. Ingest data

This topic uses ApsaraDB RDS for MySQL as an example to describe how to connect a data source to Dataphin.

Prerequisites

A data source and a project are prepared. For more information, see [Prepare for using Dataphin](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Planning** in the top navigation bar.
3. On the Planning page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source to be connected to Dataphin. Select MYSQL .
Name	The name of the data source. Set the value to dataphin.
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none">◦ If you want to use the data source only in the production environment, select Prod.◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods:<ul style="list-style-type: none">▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment.▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The Java Database Connectivity (JDBC) URL that you can use to connect to the data source. Enter a URL in the format of <code>jdbc:mysql://RDS ID.mysql.rds.aliyuncs.com:3306/dataphin</code> . Replace RDS ID with the ID of the ApsaraDB RDS for MySQL instance that you want to connect to Dataphin.
Username	The username that you can use to connect to the data source. Set the value to dataphin.
Password	The password that you set for the dataphin user when you create the ApsaraDB RDS for MySQL instance.

6. Click **Test Connection**.
7. After the data source passes the connectivity test, click **OK**.

9.3.5. Integrate data

This topic describes how to synchronize data from a data source to your project, including the procedure for creating and configuring a sync task.

Prerequisites

A data source is connected to Dataphin. For more information, see [Ingest data](#).

Procedure

1. [Log on to the Dataphin console.](#)

2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. Move the pointer over **Develop** in the top navigation bar and select **Integrated**.
5. Create an offline migration pipeline.
 - i. On the **Integrated** page, open the **Create Pipeline Development Script** dialog box by using one of the following methods:
 - In the **Script** section, move the pointer over the icon next to **Script** and select **Offline Single Pipeline**.
 - In the left-side navigation pane, move the pointer over the icon next to the project name and select **Offline Single Pipeline**.
 - ii. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

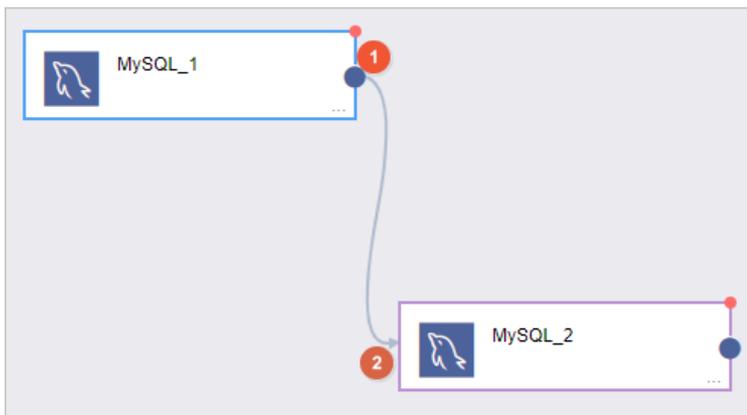
Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Select One-Time Node .
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides. Default value: offline pipeline .

- iii. Click **OK**.
6. Configure the offline migration pipeline.
 - i. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
 - ii. Click the icon before **Input**. Drag the **MySQL** icon to the pipeline canvas on the left. The **MYSQL_1** component appears on the canvas.

- iii. Right-click MySQL_1 and select **Configure Attributes**. In the MySQL Input Configuration dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ▪ The data source is of the MySQL type. ▪ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for the permission. For more information, see Data source permissions. <p>You can also click the  icon next to Data Source to go to the Planning page and connect a data source to Dataphin.</p>
Source Table	The mode for synchronizing data. Select Single Table .
Table	The name of the source table. In this example, select <code>company_sales_record_copy</code> .
Split Key	Optional. The shard key of the source table.
Input Filtering	Optional. The filter condition for input fields.
Output Fields	The output fields to be generated based on the input configuration.

- iv. Click **OK**.
- v. Click the  icon before **Output**. Drag the **Maxcompute** icon to the pipeline canvas on the left. The `Maxcompute_1` component appears on the canvas.
- vi. Drag a directed line from **MySQL_1** to **Maxcompute_1** to establish a relationship between the two components, as shown in the following figure.



- vii. Right-click Maxcompute_1 and select **Configure Attributes**. In the Maxcompute Output Configuration dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	The data source of the component. Select a data source that is bound to your project.
Table	The name of the destination table. In this example, select dataphin_test.
Loading Policy	The policy for writing data to the destination table. Select Append Data .
Parse Solution	Optional. The operations before and after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	The output fields to be generated.
Quick Mapping	<p>The GUI element that allows you to specify mappings between the input and output fields. To specify name-based mappings between the input and output fields, perform the following steps:</p> <ol style="list-style-type: none"> Move the pointer over the  icon next to Quick Mapping. Select Name-based Mapping from the drop-down list. In the Notification message, click OK.

- viii. Click **OK**.

- In the upper-left corner of the pipeline configuration tab, click **Preview** to check whether the pipeline can be run.
 - If some variables such as bizdate are configured in the components, set relevant parameters as required and click **OK** to preview the running result of the pipeline.
 - If no variable is configured in the components, you can directly preview the running result of the pipeline.
- In the upper-left corner of the pipeline configuration tab, click **Run** to synchronize data from the data source of the MYSQL_1 component to the data source that is bound to your project.
 - If some variables such as bizdate are configured in the components, set relevant parameters as required and click **OK** to run the pipeline.
 - If no variable is configured in the components, you can directly run the pipeline.
- Save, submit, and then publish the offline migration pipeline.
 - Click the  icon in the upper-right corner to save the pipeline.
 - Click the  icon in the upper-right corner to submit the pipeline. When you submit the pipeline, Dataphin checks whether you have the following permissions:
 - Read permission on the data source in each input component
 - Write permission on the data source in each output component

- iii. (Optional) Publish the pipeline.
 - If the current project is in Dev mode, publish the pipeline to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the pipeline after you submit it.

9.3.6. Define data standardization objects

Based on the methodology of dimensional modeling, you can build a bus matrix, divide data domains, and then define business processes, dimensions, atomic metrics, business filters, statistical periods, and derived metrics. This topic describes how to define data standardization objects.

Sample data

In this tutorial, statistics are collected on the total daily sales of office supplies and technical products in each province. The following table describes the fields that are selected from the `company_sales_record_copy` table for analysis.

Field	Description
<code>order_id</code>	The order ID.
<code>area</code>	The region.
<code>province</code>	The province.
<code>city</code>	The city.
<code>product_type</code>	The type of the product.
<code>customer_name</code>	The name of the customer.
<code>report_date</code>	The date of the order.
<code>order_amt</code>	The sales amount of the order.

Terms

- Dimension

A dimension, which serves as the basis of measurement, is composed of a type of business attributes. A dimension can also be called an entity object. When you divide data domains and build a bus matrix, you must define dimensions based on the analysis of business processes. In this example of analyzing the total daily sales of office supplies and technical products in each province in a year, province is used as the dimension for data modeling.

- Business process

A business process refers to a business event of an enterprise, such as order placement, payment, and refund. In most cases, a business process is an indivisible event. This tutorial uses order placement as the business process whose ID is `order_pay`.

- Atomic metric

An atomic metric is the measure for a business event. It is an indivisible metric and has a specific business meaning. You must define the `measure` and `business event` for an atomic metric. For example, the payment amount is an atomic metric, where payment is the event and amount is the measure. In this tutorial, the atomic metric `sum(order_amt)` is created based on the sum of the measure `order_amt`.

- Business filter

A business filter defines the scope of a business. A business filter is unique within a business unit and belongs to only one source logical table. The computing logic of a business filter is defined based on the fields of the source logical table model. This ensures that all metrics are created in a uniform and standardized manner. In this tutorial, `product_type='technical products'` and `product_type='office supplies'` are used as business filters.

- **Derived metric**

A derived metric is composed of the `atomic metric, business filter, statistical period, and dimension (statistic granularity)`. A derived metric is an atomic metric that is constrained by the statistical business scope. This tutorial takes the total sales amount of office supplies and technical products of the last day in each province as the derived metric, province as the dimension, order placement as the business process, the total sales amount `sum(order_amt)` as the atomic metric, technical products and office supplies as business filters, and the last day as the statistical period.

9.3.7. Create data modeling objects

This topic describes how to create data modeling objects.

Prerequisites

- A data source and a project are prepared. For more information, see [Prepare for using Dataphin](#).
- The data source is connected to Dataphin. For more information, see [Ingest data](#).
- Data synchronization is completed. For more information, see [Integrate data](#).
- Data standardization objects are defined. For more information, see [Define data standardization objects](#).

Create a dimension

1. [Log on to the Dataphin console](#).
2. Go to the **Dimensions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. (Optional) On the Develop page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. In this example, select the project that you create.
 - iv. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - v. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon.
3. In the **Dimensions** section, click the  icon in the upper-right corner.
4. On the **Create Dimension** tab, set the parameters as required.

Section	Parameter	Description
Basic Dimension Information	Data Domain	The data domain of the dimension to be created. Select <code>test_dataphin</code> .
	Dimension Name	The name of the dimension. Enter province.
	Dimension Display Name	The display name of the dimension. Enter province.

Section	Parameter	Description
	Dimension Description	The description of the dimension.
Dimension Logic	Primary Key Name	The name of the primary key. Enter province.
	Primary Key Display Name	The display name of the primary key. Enter province.
	Primary Key Type	The data type of the primary key. Select STRING.
	Primary Key Computing Logic	<p>The computing logic of the primary key. Dataphin allows you to define the primary key computing logic of a dimension by writing SQL statements.</p> <ol style="list-style-type: none"> i. Click Example next to Primary Key Computing Logic to view an example of SQL statements. Enter the following SQL statement in the SQL editor: <div style="background-color: #f0f0f0; padding: 5px; margin: 5px 0;"> <pre>select province from dataphin_test</pre> </div> ii. Click Code Check to verify whether the SQL statement that you entered is valid.

5. Save and submit the dimension.
 - i. Click the  icon in the upper-right corner to save the dimension.
 - ii. Click the  icon in the upper-right corner to submit the dimension.
6. View the corresponding logical dimension table.
 - i. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. After you create a dimension, Dataphin automatically generates a logical dimension table for the dimension. The name of the generated logical dimension table is the same as the dimension name that you defined.
 - ii. In the Logical Dimension Tables section, use the dimension name that you defined to search for the corresponding logical dimension table. Alternatively, search for the logical dimension table dim_province in the data domain test_dataphin.

Create a business process and a logical fact table

1. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon.
2. In the **Business Processes** section, click the  icon in the upper-right corner.
3. In the **Create Business Process** dialog box, set the parameters as required.

Create Business Process
✕

Business Unit Project Project Type Application Data Store

* Data Domain test_dataphin ▼

* Name order_pay

* Display Name order_pay|

Description

0/128 ↵

Cancel
Submit

Parameter	Description
Data Domain	The data domain of the business process to be created. Select test_dataphin.
Name	The name of the business process. Enter order_pay.
Display Name	The display name of the business process. Enter order_pay.

4. Click **Submit**.
5. In the dialog box that appears, enter your comments.
6. Click **OK**.
7. Create a logical fact table.
 - i. Click **order_pay** in the Business Processes section. On the View Attributes tab, you can view the details about the business process.
 - ii. Move the pointer over the i icon and select **Create Logical Table**.

iii. In the **Create Logical Fact Table** dialog box, set the parameters as required.

Parameter	Description
Business Process	The business process that corresponds to the logical fact table to be created. Select order_pay (order_pay) .
Fact Table Type	The type of the logical fact table. Select Transaction Fact Table .
Display Name	The display name of the logical fact table. Enter shishibiao .
Main Source Table	The main source table of the logical fact table. Select Project name.dataphin_test. In this example, select test_xianshang_dev.dataphin_test .

iv. Click **Next**.

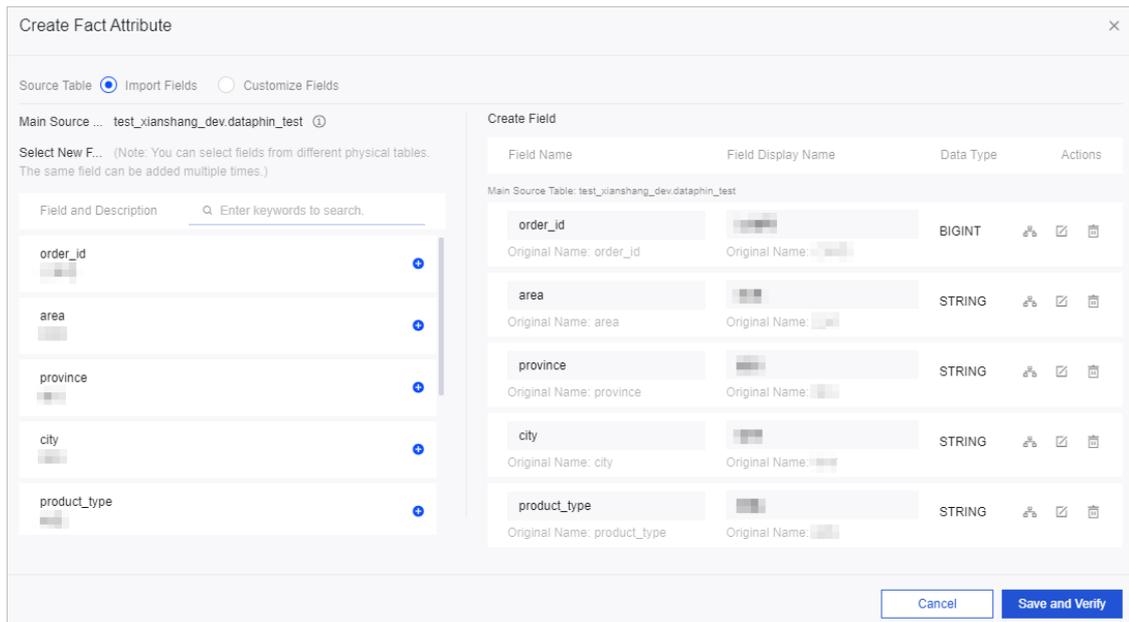
v. In the **Primary Key Definition** step, set the **Set Primary Key** parameter to **No**.

vi. Click **Submit**.

8. Configure the logical fact table.

i. On the configuration tab of the **fac_order_pay_df** logical fact table that appears, click **Add Measure** in the **Central Table** section.

ii. In the **Create Measure** dialog box, set the parameters as required.



Parameter	Description
Source Table	Specifies whether to import fields from the source table or use SQL statements to reference fields. Select Import Fields .
Create Field	The field to be added. Add a field by performing the following steps: <ol style="list-style-type: none"> Click the  icon next to a field name in the Select New Fields section. Enter a name in the Field Display Name field.

iii. Click **Save and Verify**.

iv. On the configuration tab of the fac_order_pay_df logical fact table, click **Create Dimension Association** in the Central Table section.

v. In the **Create Dimension Association** dialog box, set the parameters as required.

The screenshot shows the 'Create Dimension Association' dialog box. It contains the following fields and settings:

- Associated Dimension:** test_dataphin / dim_province province
- Dimension Type:** Common Dimension
- Primary Key and Computing Logic:** province
- Edit Association Logic:**

Fact Table Field to Be Associated	Associated With	Selected Associated Dimension
Fact Attribute / province	Associated With	dim_province.pro...
- Edit Dimension Role:**
 - Role Name: dim_ provinc
 - Role Display Name: province
- Default Value Settings:**
 - Default Value: -110

- **Associated Dimension:** Select the logical dimension table dim_province in the data domain test_dataphin.
- **Edit Association Logic:** Select province under Measure.
- Use the default values for other parameters.

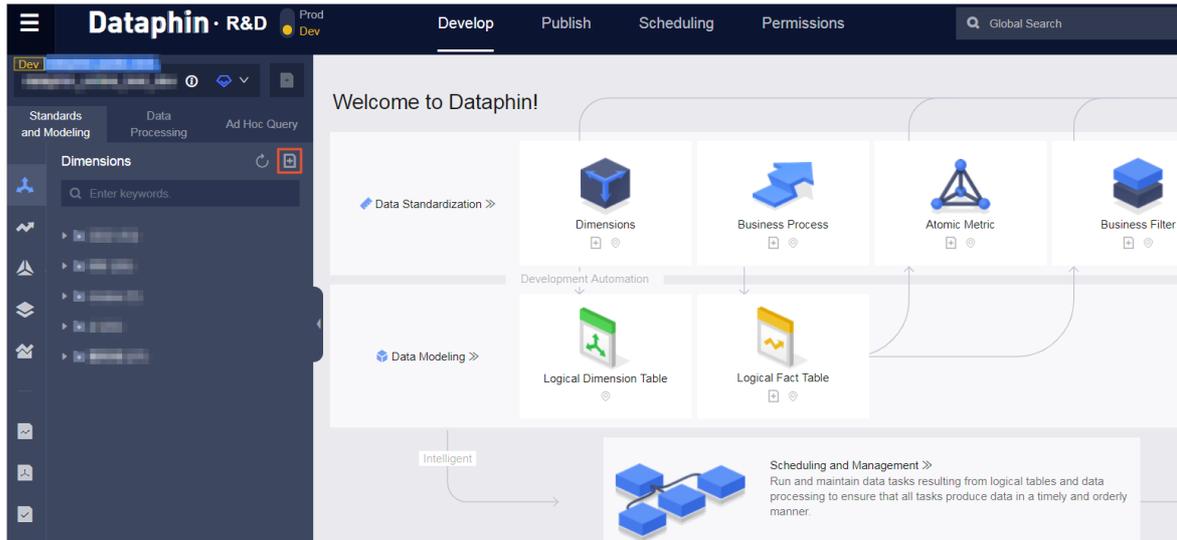
vi. Click **OK**.

9. Save and submit the logical fact table.

- Click the  icon in the upper-right corner to save the logical fact table.
- Click the  icon in the upper-right corner to submit the logical fact table.

Create an atomic metric

- On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon.
- In the **Atomic Metrics** section, click the  icon in the upper-right corner and select **Create Atomic Metric**.



3. On the **Create Atomic Metric** tab, select `fct_order_pay_df` from the **Source Table** drop-down list. In the Atomic Metrics section, click **Create Atomic Metric**.
4. In the **Create Atomic Metric** dialog box, set the parameters as required.

Create Atomic Metric
✕

* Primary Source Field ?

* Name * Display Name

Description 0/128

Data Type

Statistical Period Indicator ?

* Computing Logic ? Accumulable: Yes No

Beautify
Example
Code Check
?

```
1 sum (fct_order_pay_df.order_amt)
```

Parameter	Description
Primary Source Field	The source field of the atomic metric to be created. Select <code>fct_order_pay_df.order_amt</code> .
Name	The name of the atomic metric. Enter <code>order_amt</code> .
Display Name	The display name of the atomic metric. Enter <code>order_amt</code> .
Data Type	The data type of the atomic metric. Select <code>BIGINT</code> .
Field	The field of the statistical period. Select <code>report_date</code> .

> Document Version: 20210621

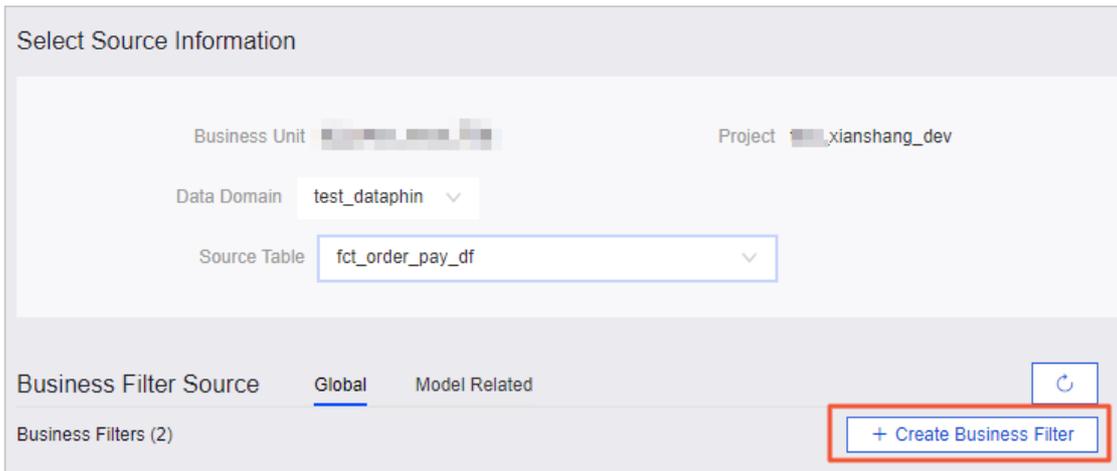
2425

Parameter	Description
Computing Logic	The computing logic of the atomic metric. Enter <code>sum(fct_order_pay_df.order_amt)</code> .

- Click **Code Check** to verify whether the SQL statement that you entered is valid.
- After the SQL statement passes the code check, click **Submit**.
- In the dialog box that appears, enter your comments.
- Click **OK**.

Create a business filter

- On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon.
- In the **Business Filters** section, click the  icon in the upper-right corner.
- On the **Create Business Filter** tab, select `fct_order_pay_df` from the **Source Table** drop-down list. In the Business Filter Source section, click **Create Business Filter**.



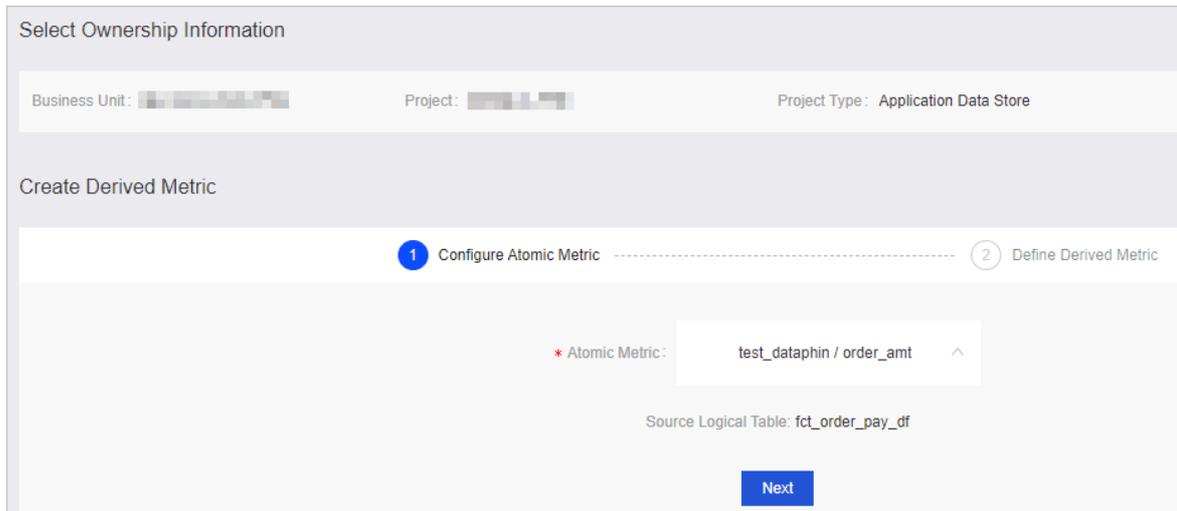
- In the **Create Business Filter** dialog box, set the parameters as required.

Parameter	Description
Primary Source Field	The source field of the business filter to be created. Select <code>fct_order_pay_df.product_type</code> .
Name	The name of the business filter. Enter <code>product_type_bangongyongpin</code> .
Display Name	The display name of the business filter. Enter <code>product_type_office supplies</code> .
Computing Logic	The computing logic of the business filter. Enter <code>fct_order_pay_df.product_type='office supplies'</code> .

- Click **Code Check** to verify whether the SQL statement that you entered is valid.
- After the SQL statement passes the code check, click **Submit**.
- In the dialog box that appears, enter your comments.
- Click **OK**.

Create a derived metric

1. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon.
2. In the **Derived Metrics** section, click the  icon in the upper-right corner.
3. On the **Create Derived Metric** tab, select `order_amt` under `test_dataphin` from the **Atomic Metric** drop-down list.



Select Ownership Information

Business Unit: [redacted] Project: [redacted] Project Type: Application Data Store

Create Derived Metric

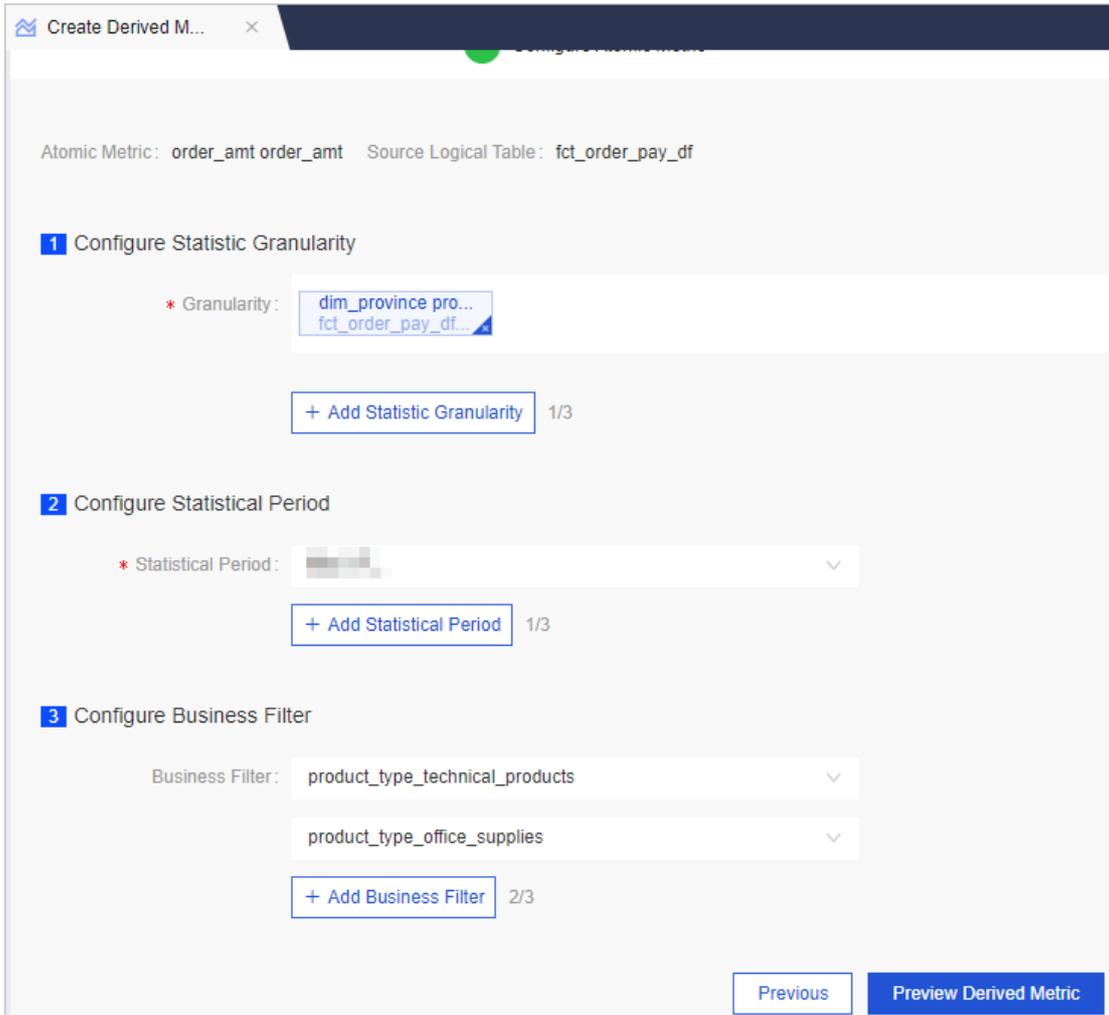
1 Configure Atomic Metric 2 Define Derived Metric

* Atomic Metric: test_dataphin / order_amt ^

Source Logical Table: fct_order_pay_df

Next

4. Click **Next**.
5. In the **Define Derived Metric** step, set the parameters as required.



Parameter	Description
Granularity	The statistic granularity of the derived metric to be created. Select fct_order_pay_df.dim_province under dim_province province.
Statistical Period	The statistical period of the derived metric. Select the last day.
Business Filter	The business filter of the derived metric. Select product_type_office supplies and product_type_technical products.

6. Click **Preview Derived Metric**.
7. In the **Change Derived Metric** section, confirm the parameter settings and click **Submit**.
8. Submit the derived metric.
 - i. Click **Submit** to submit the derived metric.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.

9.3.8. Generate retroactive data

To verify that code and tasks are valid, you can initiate a retroactive data generation task and verify the data. This topic describes how to generate retroactive data for logical table conversion tasks and sync tasks.

Prerequisites

Data modeling objects are created. For more information, see [Create data modeling objects](#).

Context

After you create and submit a dimension, a logical fact table, and a logical aggregate table, Dataphin automatically generates the corresponding logical table conversion tasks, for example, `dim_province_core_od001_v1`, `fct_order_pay_df_od000_v1`, and `dws_province_od000_v1`. You can use the same method to generate retroactive data for these logical table conversion tasks. This topic describes how to generate retroactive data for the logical table conversion task that is generated after you create a dimension, namely, `dim_province_core_od001_v1`.

Generate retroactive data for a logical table conversion task

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Table Conversion Tasks** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select the project that you create.
 - iii. In the top navigation bar, click **Scheduling**.
 - iv. In the left-side navigation pane, click the **Global Management** tab.
 - v. On the **Global Management** tab, move the pointer over the left-side navigation submenu and click the  icon.
 - vi. In the **Recurring Tasks** section, click the **Logical Table Conversion Tasks** tab.
3. Click the logical table conversion task `dim_province_core_od001_v1`.
4. In the directed acyclic graph (DAG), right-click the current node and select **Generate Retroactive Data**.
5. In the **Generate Retroactive Data** dialog box, set the **Data Timestamp** parameter to 2019-10-27~2019-10-27. Set the **Select Downstream Nodes** parameter to **No**.

 **Note** The system automatically generates a value for the **Instance Name** parameter when you open the **Generate Retroactive Data** dialog box. You can change the value.

6. Click **OK**.
7. View the running result of the retroactive data generation instance.
 - i. On the **Global Management** tab, move the pointer over the left-side navigation submenu and click the  icon.
 - ii. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance. In the DAG, right-click the current node and select **View Operations Log**.
 - iii. In the **Operations Log** pane, view the operational logs of the instance.

Generate retroactive data for a sync task

1. On the **Global Management** tab, move the pointer over the left-side navigation submenu and click the  icon.
2. In the **One-time Tasks** section, click the one-time task **dataphin**.
3. On the page that appears, click **Run** in the upper-right corner.

4. In the **Run** dialog box, set the **Data Timestamp** parameter to `2019-10-27`.
5. Click **OK**.
6. View the running result of the one-time instance.
 - i. On the **Global Management** tab, move the pointer over the left-side navigation submenu and click the  icon.
 - ii. In the **One-time Task Instances** section, click the instance that is generated for the one-time task **dataphin**.
 - iii. On the page that appears, click **View Operational Log** in the top navigation bar. On the **Operational Log** page, view the operational logs of the instance.

9.3.9. Verify data

This topic describes how to use an ad hoc query task to verify that your expected data is generated.

Prerequisites

Retroactive data is generated for the required logical table conversion tasks and sync tasks. For more information about how to generate retroactive data, see [Generate retroactive data](#).

Context

The ad hoc query feature allows you to perform theme-based data query. MaxCompute SQL statements can be executed when MaxCompute is selected as the computing engine type of Dataphin. Dataphin automatically identifies and switches the SQL type based on the computing engine type that you set.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Ad Hoc Query** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select the project that you create.
 - iii. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
3. In the **Ad Hoc Query** section, click the  icon next to **Ad Hoc Query**.
4. In the **Create Item** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the ad hoc query task to be created. Enter <code>dim_province</code> .
Description	The description of the ad hoc query task.
Select Directory	The folder where the ad hoc query task resides.

5. Click **OK**.
6. On the **Code Editor** tab of the ad hoc query task, enter `select * from dataphin_test where ds='20191027'`.
7. Save and run the ad hoc query task.
 - i. Click the  icon in the upper-right corner to save the ad hoc query task.

- ii. Click the  icon in the upper-right corner to run the ad hoc query task.
8. After the SQL statement is executed, view the running result on the **Result** tab.



	order_id	area	province	city	product_type	customer_name	report_date	order_amt	ds
1	13729				office supplies		2013-01-01 00:00:00	872.48	20191027
2	28774				office supplies		2013-01-01 00:00:00	180.36	20191027
3	37537				technical products		2013-01-02 00:00:00	4083.19	20191027
4	37537				technical products		2013-01-02 00:00:00	4902.38	20191027
5	37537				office supplies		2013-01-02 00:00:00	1239.06	20191027
6	44069				technical products		2013-01-02 00:00:00	137.63	20191027

9.3.10. Publish data

This topic describes how to publish data from the development environment to the production environment for scheduling.

Prerequisites

The data to be published is verified. For more information, see [Verify data](#).

Context

You can publish data only when the data is generated in a project in Dev mode.

- If data is generated in a project in Basic mode, relevant tasks can be scheduled in the production environment after you submit the data.
- If data is generated in a project in Dev mode, you must publish the data to the corresponding project in Prod mode for scheduling after you submit the data. The data to be published may be data standardization and data modeling objects or data processing tasks.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the Develop page, click the icon next to the project name in the upper-left corner and select the project that you create.
4. In the top navigation bar, click **Publish**.
5. Publish data standardization and data modeling objects.
 - i. On the **Objects to Publish** page, click the **Standards and Modeling** tab.
 - ii. On the **Standards and Modeling** tab, select the objects that you develop in this tutorial.
 - iii. Click **Publish** in the lower part of the page.
 - iv. In the **Publish** dialog box, set the **Publishing Name** parameter and click **OK**. You can use the default value of Publishing Name.
6. Publish batch processing tasks.
 - i. On the **Objects to Publish** page, click the **Batch Processing** tab.
 - ii. On the **Batch Processing** tab, select the sync task that you develop in this tutorial.
 - iii. Click the  icon in the **Actions** column.
 - iv. Alternatively, click **Publish** in the lower part of the page.

- v. In the **Publish** dialog box, set the **Publishing Name** parameter and click **OK**. You can use the default value of Publishing Name.
7. View publishing records.
 - i. In the left-side navigation pane, click **Publishing History**. The Publishing History page appears.
 - ii. Click the **Standards and Modeling** tab and view publishing records of data standardization and data modeling objects.
 - iii. Click the **Batch Processing** tab and view publishing records of data processing tasks.

9.3.11. Verify the scheduling result

This topic describes how to verify that the scheduling result meets your expectations.

Context

After you publish data to the production environment, relevant tasks will not be run until the next day. You can simulate the task scheduling in the development environment to view the scheduling result.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. Go to the **Global Management** tab.
 - i. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select the project that you create.
 - ii. In the top navigation bar, click **Scheduling**. The **Global Management** tab appears.
4. Generate retroactive data for one-time tasks and logical table conversion tasks of logical dimension tables, logical fact tables, and logical aggregate tables. For more information, see [Generate retroactive data](#).
5. Go to the **Ad Hoc Query** tab.
 - i. In the top navigation bar, move the pointer over **Develop** and select **Develop**. The **Develop** page appears.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
6. Create an ad hoc query task.
 - i. On the **Ad Hoc Query** tab, click the  icon.

ii. In the **Create Item** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the ad hoc query task to be created. Enter dws_province.
Description	The description of the ad hoc query task.
Select Directory	The folder where the ad hoc query task resides.

iii. Click **OK**.

iv. On the **Code Editor** tab of the ad hoc query task, enter `select * from dataphin_test where ds='20191027'`.

7. Save and run the ad hoc query task.

i. Click the  icon in the upper-right corner to save the ad hoc query task.

ii. Click the  icon in the upper-right corner to run the ad hoc query task.

8. After the SQL statement is executed, view the running result on the **Result** tab.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
province																
order_amt_1d_product_type	W	W	W	W	W	W	W	W	W	764	W	W	110	W	W	W
order_amt_1d_product_type_jishuchanping	W	W	W	3443	W	W	W	W	W	W	W	25	W	W	W	W
ds	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028	20191028

9.4. Management Center

9.4.1. Initialize metadata

After the compute cluster information is obtained, you can log on to the Dataphin console as the operations and maintenance (O&M) super administrator to configure the metadata warehouse. This topic uses Apsara Stack Dataphin whose computing engine type is MaxCompute as an example.

Procedure

1. [Log on to the Dataphin console](#) as the O&M super administrator. For more information about the O&M super administrator account and password, see [Instructions for the system administrator](#).

Note System administrators must keep the account and password of the O&M super administrator strictly confidential and perform operations with caution after logging on as the O&M super administrator.

2. On the Dataphin homepage, click **Management Center** in the top navigation bar. In the left-side navigation pane, click **Metadata Warehouses Configuration**. On the page that appears, click **Start**.
3. In the **Select Computing Engine** step, select a computing engine type and click **Next**. In this example, select **MaxCompute v1.1**.
4. In the **Parameter Configuration** step, set the parameters correctly, as shown in the following figure. Click **Test Connection**. After the test is passed, click **Next**.

The screenshot shows a configuration form with the following fields and values:

- Endpoint: `${aliyun.odps.endpoint}`
- Project Name: `${dataphin.odps.project}`
- Access ID: `${dataphin.odps.account.accessid}`
- Access Key: `${dataphin.odps.account.accessKey}`
- Meta Project: `dataphin_mc_meta`

Buttons at the bottom right: **Test Connection**, **Previous**, **Next**.

5. Perform the subsequent steps as prompted.

Note The subsequent steps, including configuring MaxCompute at the backend, take about 15 minutes. If a message appears to inform you that the subsequent steps are successfully performed, click **Complete**.

9.4.2. Manage members

This topic describes how to add and remove members and configure contact information for members in Dataphin.

Go to the Members Management page

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Management Center** in the top navigation bar.
3. On the **Members Management** page, different roles including members and the super administrator have different permissions:
 - o If you are a member, you can view the following information of all members: username, display name, and when a member is added to the system. However, you can only view and edit your own mobile number, email address, and webhook URL.
 - o If you are the super administrator, you can perform the following operations:

- View the following information of all members: username, display name, and when a member is added to the system. You can also click the  icon in the **Mobile Phone**, **Mailbox**, and **DingTalk Group Chatbot** columns to view the mobile number, email address, and webhook URL of a member. After you view the mobile number, email address, and webhook URL of a member, you can click the  icon in the **Mobile Phone**, **Mailbox**, and **DingTalk Group Chatbot** columns to hide corresponding information. You can also edit the mobile number, email address, and webhook URL for all members.
- Click the  icon in the headings of the **Mobile Phone**, **Mailbox**, and **DingTalk Group Chatbot** columns to view the mobile numbers, email addresses, and webhook URLs, if configured, of all members. After you view the mobile numbers, email addresses, and webhook URLs of all members, you can click the  icon in the headings of the **Mobile Phone**, **Mailbox**, and **DingTalk Group Chatbot** columns to hide corresponding information of all members.

Add members

 **Note** Only the super administrator can add members to Dataphin.

- On the **Members Management** page, click **Add Members** in the upper-right corner.
- In the **Add Members** dialog box, click the  icon next to the **Username** parameter.
- Select a RAM user and click **OK**.

You can add multiple RAM users as members at a time. You can enter a keyword in the search box to search for RAM users you want to add.

Configure contact information for members

 **Note** The super administrator can configure contact information for one or more members at a time. Members can configure only their own contact information.

To configure contact information for one or more members at a time, perform the following steps:

- On the **Members Management** page, select one or more members and click **Edit Members** in the upper-right corner.
 - In the **Modify Contact Information** dialog box, set the parameters as required.

Parameter	Description
Mobile Phone	The mobile number of the member.  Note Only mobile numbers in mainland China can be added.
Mailbox	The email address of the member.
DingTalk Group Chatbot	The webhook URL of the member. Enter <code>https://oapi.dingtalk.com/robot/send?access_token=713697da1c2a874bc80518e1872e9627d7e79d3015dd5791a1f03589be76ce3f</code> .

- Click **OK**.

- i. On the **Members Management** page, find the member for whom you want to configure contact information and click the  icon in the **Actions** column.
- ii. In the **Modify Contact Information** dialog box, set the parameters as required.

Parameter	Description
Mobile Phone	The mobile number of the member.  Note Only mobile numbers in mainland China can be added.
Mailbox	The email address of the member.
DingTalk Group Chatbot	The webhook URL of the member. Enter <code>https://oapi.dingtalk.com/robot/send?access_token=713697da1c2a874bc80518e1872e9627d7e79d3015dd5791a1f03589be76ce3f</code> . You can move the pointer over the  icon to view how to receive alert notifications by using a DingTalk chatbot.

- iii. Click **OK**.

Remove members

 **Note** Only the super administrator can remove members from Dataphin.

To remove one or more members at a time, perform the following steps:

- i. To remove multiple members at a time, select the members and click **Delete** in the upper-right corner.
- ii. In the message that appears, click **OK**.
- i. To remove a single member, find the member and click the  icon in the **Actions** column.
- ii. In the message that appears, click **OK**.

 **Note** By default, only RAM users who are added as members can be removed from Dataphin. The super administrator cannot be removed.

Synchronize account information from another account system to Dataphin

If the account information is updated in the RAM user list, you can click **Account System Synchronization** on the **Members Management** page to synchronize the updated account information to Dataphin.

9.4.3. Configure the computing engine settings

This topic describes how to configure the computing engine settings of Dataphin.

Context

Dataphin supports the computing engines for both stream processing and batch processing.

- By default, Dataphin sets **Flink** as the computing engine type for stream computing.
- Dataphin supports **MaxCompute** and **Hadoop-2.6** as the computing engine type for batch computing.

Procedure

1. [Log on to the Dataphin console](#) as the super administrator.
2. On the Dataphin homepage, click **Management Center** in the top navigation bar.
3. On the page that appears, click **Computing Engine Configuration** in the left-side navigation pane.
4. On the **Computation Configuration** page, select the computing engine type for batch processing.
5. Set the parameters as required.
 - If you select **MaxCompute** as the computing engine type, you must set the **Endpoint** parameter. Click **Verify**. After the setting passes the verification, click **Submit**.
 - If you select **Hadoop-2.6** as the computing engine type, you must set the **NameNode** parameter. Click **Verify**. After the setting passes the verification, click **Submit**.

9.4.4. Configure the intelligent data engine

9.4.4.1. Configure data skew optimization

This topic describes how to use the data skew optimization feature of Dataphin to optimize tasks in an abnormal status during scheduling.

Background information

The data skew optimization feature can be used in the following scenarios during scheduling:

1. Assume that you enable the data skew optimization feature and submit a task for the first time. If the duration and processed data volume of the task exceed the specified duration threshold, multiple of the average duration, and multiple of the average data volume, the intelligent data engine detects data to analyze the task and determine whether the task is abnormal.
2. When the task is being run, the intelligent data engine detects data for the task to find hotspot data in associated fields. Based on the detection result, the engine refactors the code of the ongoing task to avoid data skew. In addition, the engine terminates the ongoing task and submits new code to rerun the task. This optimizes logical table tasks.
3. The intelligent data engine records the skew state to detect data on the next day. If data skew still occurs in the detection result on the next day, the engine continues to use the optimized code. If no data skew occurs in the detection result, the engine uses the code without data skew optimization.
4. For a rerun logical table task:
 - If the table that is referenced by the task has been detected and the detection result shows that the fields are skewed, the intelligent data engine optimizes the submitted SQL statements based on the detection result, and uses the optimized code to run the task.
 - If the table that is referenced by the task has not been detected, the intelligent data engine detects data for the task and refactors the code based on the detection result in the same way as that for a task that is submitted for the first time. If data skew occurs in the tasks and models that are automatically generated by the intelligent data engine during production, the intelligent data engine processes the exceptions.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Management Center** in the top navigation bar.
3. On the page that appears, click **Intelligent Data Engine** in the left-side navigation pane. The **Data Skew Optimization** tab appears.

4. Set the parameters as required. The following table describes the parameters.

Parameter	Description	Example
Data Skew Optimization	Specifies whether to enable the data skew optimization feature for the intelligent data engine.	Enable
Duration Threshold	The threshold of the execution duration of a task. If the duration of a task exceeds this threshold, the intelligent data engine detects data to analyze the task and determine whether the task is abnormal. Unit: milliseconds.	600000
Multiple of Average Duration	The multiple of the average duration of all tasks. If the duration of a task is longer than the specified multiple of the average duration of all tasks, the intelligent data engine detects data to analyze the task and determine whether the task is abnormal.	2
Multiple of Data Amount	The multiple of the average data volume processed by all tasks. If the processed data volume of a task is greater than the specified multiple of the average data volume processed by all tasks, the intelligent data engine detects data to analyze the task and determine whether the task is abnormal.	2
Value Threshold	The threshold that is used to identify skewed data. If a value in a dataset is greater than this threshold, the value is identified as a skewed value.	1000000
Multiple of Average Value Occurrences	The multiple of the average occurrences of all values in a dataset. If the frequency of occurrence of a value in a dataset is higher than the specified multiple of the average occurrences of all values in the dataset, the value is identified as a skewed value.	2

5. After you set the preceding parameters, click **OK**. In the message that appears, click **OK**.

To restore the parameters on the **Data Skew Optimization** tab to default values, click **Restore Defaults**. In the message that appears, click **OK**.

9.4.4.2. Configure code optimization

This topic describes how to use the code optimization feature of Dataphin to simplify production code.

Background information

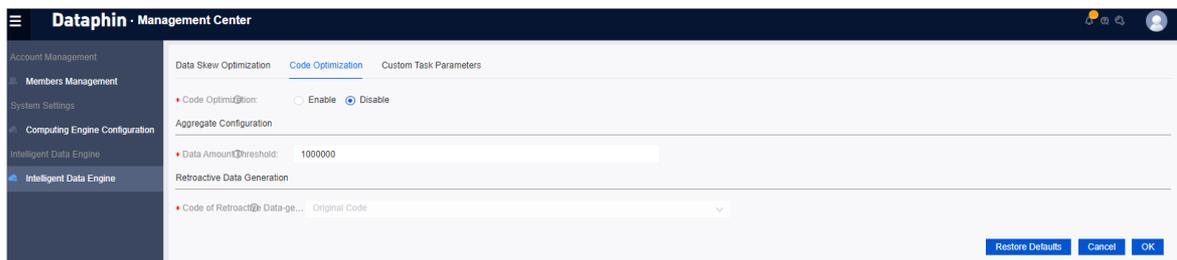
You can enable the code optimization feature to reuse the semantics of SQL statements and optimize production code. This feature can be used to optimize only code that involves derived metrics. For example, the total payment amount of each commodity category in the last 30 days can be calculated by adding up the daily payments of products under the category during this period.

 **Note** The dimensions that are associated with a derived metric may vary with time.

For example, an incremental fact table that records order payments is associated with the commodity dimension, and the commodity dimension is associated with the category dimension. Based on this fact table, you can calculate the total payment amount of each category in the last 30 days. Assume that the category of a product changed from category1 to category2 on a day of the last 30 days. Before the code optimization feature is enabled, all payments of the product in the last 30 days are counted as payments under category2. After the code optimization feature is enabled, the payments received before the category of the product changed are counted as payments under category1 and the payments received after the category of the product changed are counted as payments under category2.

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Management Center** in the top navigation bar.
3. On the page that appears, click **Intelligent Data Engine** in the left-side navigation pane.
4. On the **Intelligent Data Engine** page, click the **Code Optimization** tab.



5. On the **Code Optimization** tab, set the following parameters:
 - **Code Optimization:** specifies whether to enable the code optimization feature. Valid values: **Enable** and **Disable**.
 - **Data Amount Threshold:** the threshold of the data volume in a logical table for triggering code optimization. If the data volume of a logical table exceeds this threshold, code optimization is triggered to enable the intelligent data engine to refactor the code and optimize data computing.
 - **Code of Retroactive Data-generated Task:** the code to be run to generate retroactive data for historical tasks before the code is optimized. Code optimization is triggered based on the analysis of derived metrics and logical aggregate tables. To generate retroactive data for historical tasks before the code is optimized, you must select the code to be run. You can select the original code or optimized code. You cannot change the value of this parameter.
6. After you set the preceding parameters, click **OK**. In the message that appears, click **OK**.
To restore the parameters on the **Code Optimization** tab to default values, click **Restore Defaults**. In the message that appears, click **OK**.

9.4.4.3. Set custom task parameters

This topic describes how to set custom task parameters to prevent improper resource allocation in Dataphin.

Context

Custom task parameters are used to specify some parameters of computing engines. This ensures that all computing engines that your enterprise uses can adapt to Dataphin and also improves the efficiency of logical table tasks.

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Management Center** in the top navigation bar.

3. On the page that appears, click **Intelligent Data Engine** in the left-side navigation pane.
4. On the **Intelligent Data Engine** page, click the **Custom Task Parameters** tab.
5. On the **Custom Task Parameters** tab, set the parameters as required.

Parameter	Description
Custom Parameters	Specifies whether to enable custom task parameters. Valid values: Enable and Disable .
Parameter Configuration	<p>The custom task parameters you set for computing engines that you create in Dataphin. These parameters must be consistent with the settings of the computing engines. Example:</p> <pre>set odps.sql.mapper.cpu=100 set odps.sql.mapper.split.size=256 set odps.sql.reducer.cpu=100 set odps.sql.joiner.cpu=100</pre> <p> Note Custom task parameters take effect based on the priority. The logical table conversion task settings of a logical table task take precedence over the settings of these custom task parameters.</p>

6. After you set the preceding parameters, click **OK**.
7. In the message that appears, click **OK**.
8. (Optional) Restore the parameters to default values.
 - i. On the **Custom Task Parameters** tab, click **Restore Defaults**.
 - ii. In the message that appears, click **OK**.

9.5. Notifications

9.5.1. View and handle messages

This topic describes how to view and handle messages in Message Center.

View and handle unread messages

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, move the pointer over the  icon in the top navigation bar and select **Notifications**. In the **Notifications** pane, you can view some messages on the **Messages** tab.
3. In the **Notifications** pane, click **View All** in the lower part.
4. On the page that appears, click **Unread Messages** under **Message Center** in the left-side navigation pane.
5. On the **Unread Messages** page, you can click the **Permission Management**, **Process Control**, and **Data Governance** tabs to view corresponding messages.
6. On the **All** tab, you can mark unread messages as read by using one of the following methods:
 - Find an unread message and click the  icon in the **Actions** column.
 - Select multiple unread messages and click **Mark as Read** in the lower part of the page.

- To mark all unread messages as read, click **Mark All as Read** in the lower part of the page.

View and handle all messages

1. In the left-side navigation pane, click **All Messages** under **Message Center**.
2. On the **All Messages** page, you can click the **Permission Management**, **Process Control**, and **Data Governance** tabs to view corresponding messages, either unread or read.
3. On the **All** tab, you can mark unread messages as read by using one of the following methods:
 - Find an unread message and click the  icon in the **Actions** column.
 - Select multiple unread messages and click **Mark as Read** in the lower part of the page.
 - To mark all unread messages as read, click **Mark All as Read** in the lower part of the page.

9.5.2. View and handle tasks

Task Center allows you to manage all process-related tasks in Dataphin. This topic describes how to view and handle tasks in Task Center.

Context

This topic uses the permission application and approval process as an example. This process consists of the following actions:

1. Submit an application.
2. Wait for approval.
3. Approve, reject, or transfer the application, or add an approver.
4. Withdraw the application.

View and handle unprocessed tasks

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, move the pointer over the  icon in the top navigation bar and select **Notifications**. In the **Notifications** pane, you can view some tasks on the **Tasks** tab.
3. In the **Notifications** pane, click **View All** in the lower part. The **Unprocessed Tasks** page under **Task Center** appears.
4. On the **Unprocessed Tasks** page, you can view the tasks that need to be handled. You can find unprocessed tasks by using one of the following methods:
 - Click **Permission Approval** next to **Task Type** to filter the tasks that require you to approve permission applications.
 - Enter a keyword of the task name in the search box to search for corresponding unprocessed tasks.
5. On the **Unprocessed Tasks** page, click the name of a task in the **Task Name** column.
6. Handle the unprocessed task.

- i. On the page that appears, you can view details of the task. In the **Task Processing** section, set the parameters as required.

Parameter	Description
Approval Result	<p>The approval result of the task. Valid values:</p> <ul style="list-style-type: none">▪ Approve: approves the application in the task and grants permissions to the applicant.▪ Reject: rejects the application in the task and requests the applicant to apply again or modify the application content.▪ Transfer To: transfers the task to another person. You can select a person to whom the task is transferred. After you transfer the task, the selected person is responsible for processing the application in the task.▪ Add Approver: adds an approver. You can select a person as a new approver.<ul style="list-style-type: none">▪ After the new approver approves the application in the task, the task will be transferred back to you.▪ If the new approver rejects the application in the task, the process ends.
Comments	The comments of the approval.

- ii. After you set the preceding parameters, click **Submit**.

View processed tasks

1. In the left-side navigation pane, click **Processed Tasks** under **Task Center**.
2. On the **Processed Tasks** page, you can view the processed tasks. You can find processed tasks by using one of the following methods:
 - Set the **Task Type** and **Status** parameters to filter processed tasks.
 - Enter a keyword of the task name in the search box to search for corresponding processed tasks.
3. On the **Processed Tasks** page, click the name of a task in the **Task Name** column to go to the details page of the task.
4. After you view the task details, click **Cancel** in the lower part of the page.

View tasks that you initiated

1. In the left-side navigation pane, click **Initiated Tasks** under **Task Center**.
2. On the **Initiated Tasks** page, you can view the tasks you initiated. You can find initiated tasks by using one of the following methods:
 - Set the **Task Type** and **Status** parameters to filter corresponding tasks you initiated.
 - Enter a keyword of the task name in the search box to search for corresponding tasks you initiated.
3. On the **Initiated Tasks** page, click the name of a task in the **Task Name** column to go to the details page of the task.
4. After you view the task details, click **Cancel** in the lower part of the page.

9.6. Alert Center

9.6.1. Overview

Alert Center is a monitoring and alerting module in Dataphin. It displays alert events, push records, and shift schedules of the Data Quality, Stream Processing, and API Service modules.

Background information

As Dataphin supports more and more features in data collection, data warehouse building, data management, and data application, it also needs to support more complex alert scenarios.

In earlier versions, Dataphin has been able to monitor errors, latency, and timeout exceptions of scheduling tasks and generate alerts when specific conditions are met. However, you can select only a limited number of alert causes to configure alert rules. In this situation, you need a monitoring and alerting center that provides enhanced monitoring capabilities and more monitoring scenarios.

Features

Alert Center provides the following pages:

- **Events:** This page displays alert events of the Stream Processing, Data Quality, and API Service modules.
- **Push Records:** This page displays records about alert messages that Alert Center pushes.
- **Shift Schedules:** This page displays shift schedules that are arranged for alert recipients who need to handle alerts.

[Log on to the Dataphin console.](#) On the Dataphin homepage, click the  icon in the upper-right corner to go to Alert Center.

9.6.2. Terms

This topic describes terms related to Alert Center, including alert event, push record, shift schedule, and push method.

Term	Description
Alert event	An event that is generated based on the specific alert object and alert rule.
Push record	A record that is generated after Alert Center pushes an alert message to the message subscribers.
Shift schedule	A schedule that is arranged for alert recipients who need to handle alerts.
Push method	A method that Alert Center uses to push an alert message to the message subscribers.
Do-Not-Disturb	A feature that allows you to specify a period during which Alert Center does not push any alert messages to you. For example, if Alert Center repeatedly pushes alert messages of the same alert event to you, you can specify a Do-Not-Disturb period for the alert event.
Service	A module where an alert event was triggered. Currently, alert events may be triggered in the Data Quality, Stream Processing, and API Service modules.

9.6.3. Events

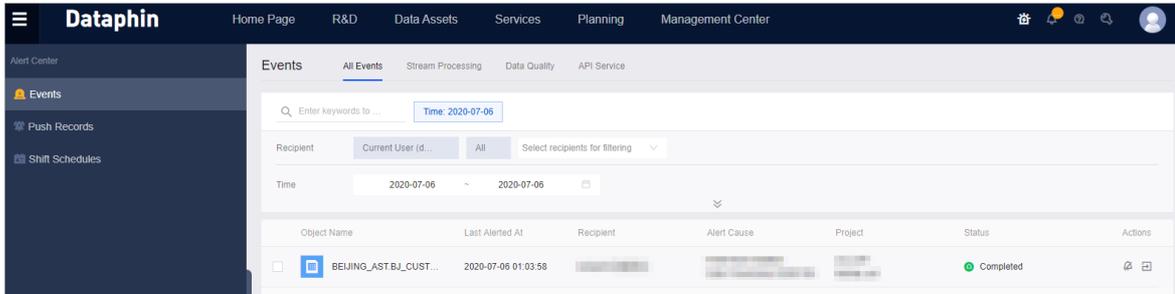
The Events page displays alert events of the Stream Processing, Data Quality, and API Service modules. This topic describes how to view and handle alert events.

Go to the Events page

1. Log on to the Dataphin console.
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to Alert Center.

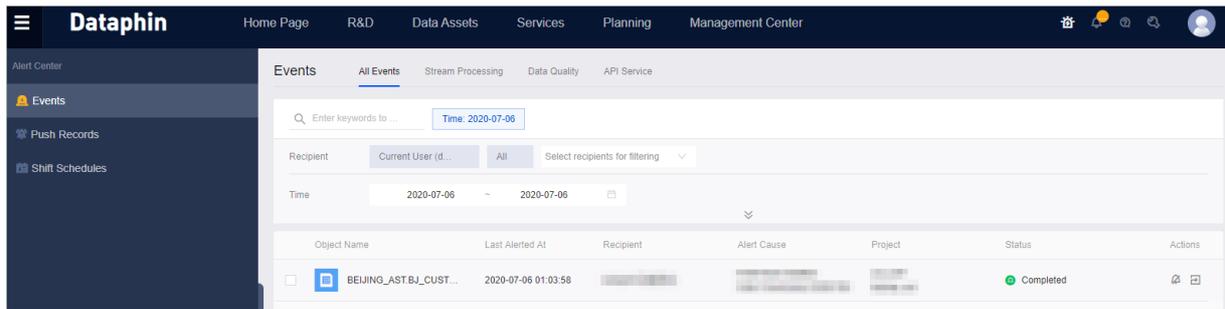
On the Events page, you can click the **Stream Processing**, **Data Quality**, or **API Service** tab to view alert events of the corresponding module.

To view all alert events of the **Stream Processing**, **Data Quality**, and **API Service** modules, click the **All Events** tab. This topic describes alert events on the **All Events** tab.



View alert events

On the **All Events** tab, you can set the **Recipient**, **Time**, **Status**, **Business Unit**, and **Project** parameters to filter alert events. You can also enter a keyword in the search box to search for alert events.



The **All Events** tab displays the **Object Name**, **Last Alerted At**, **Recipient**, **Alert Cause**, **Project**, and **Status** parameters of each alert event. The **Actions** column of an alert event lists the operations that you can perform on the alert event.

Parameter	Description
Object Name	<p>The name of the object that triggered the alert event. Alert events of each module are named based on the following conventions:</p> <ul style="list-style-type: none"> • Stream Processing: Each alert event is named after the corresponding stream processing task that is published to the production environment, for example, <code>flink_merge</code>. • API Service: Each alert event is named after the corresponding API operation, for example, <code>user_info_service</code>. • Data Quality: <ul style="list-style-type: none"> ◦ An alert event that is triggered by a physical table is named in the <code>{Project name}.{Table name}</code> format, for example, <code>v27_ast.ods_user_info</code>. ◦ An alert event that is triggered by a logical table is named in the <code>{Business unit name}.{Table name}</code> format, for example, <code>LD_demo.dws_all</code>.
Last Alerted At	The time when Alert Center pushed the last alert message for the alert event.

Parameter	Description
Recipient	<p>The recipient to whom Alert Center pushed alert messages for the alert event. You can configure multiple recipients for an alert event, including:</p> <ul style="list-style-type: none"> The owner of the object that triggers the alert event. Custom users. Users in a shift schedule.
Alert Cause	<p>The cause of the alert event. The cause of an alert event varies based on the module where the alert event may be triggered.</p> <ul style="list-style-type: none"> In Data Quality, an alert event may be triggered due to the following causes: Table Rule Violation, Field Rule Violation, and Custom Rule Violation. In API Service, an alert event may be triggered due to the following causes: Unexpected Average Response Time, Unexpected Number of Calls, Unexpected Error Rate, and Unexpected Percentage of Offline Tasks. In Stream Processing, an alert event may be triggered due to the following causes: Excessively Long Delay, Exceeds Maximum TPS, Exceeds Specified Failure Frequency, and Exceeds Specified Data Retention Period.
Project	The project to which the alert event belongs.
Status	<p>The status of the alert event. Valid values:</p> <ul style="list-style-type: none"> Completed: The alerting process is completed for the alert event. Alerting: The frequent alerting mode is set for the alert event, and the alerting start time has arrived but the end time has not. Do-Not-Disturb: You can change the status of ongoing and completed alert events to Do-Not-Disturb. Alerting (Silence Period): Alert Center does not push alert messages for the alert event because the alert time does not arrive. Assume that API Service requires Alert Center to push alert messages for each alert event at an interval of 5 minutes. After an alert message is pushed, Alert Center does not push alert messages for the same alert event within 5 minutes even if metrics that trigger the alert event are collected every second.
Actions	<p>The operations that you can perform on the alert event.</p> <ul style="list-style-type: none"> To handle an alert event, find the alert event and click the  icon in the Actions column. For more information, see Handle alert events. To change the status of an alert event to Do-Not-Disturb, find the alert event and click the  icon in the Actions column. For more information, see Change the status of alert events to Do-Not-Disturb.

View details of an alert event

On the **All Events** tab, click a name in the **Object Name** column. The **Event Details** pane appears.

GUI element	Description
Do-Not-Disturb	Allows you to change the status of the alert event to Do-Not-Disturb . For more information, see Change the status of alert events to Do-Not-Disturb .
Handle Alert	Allows you to handle the alert event. For more information, see Handle alert events .

GUI element	Description
Object Type	<p>The type of the object that triggered the alert event. Valid values of the Object Type parameter of an alert event vary based on the module where the alert event may be triggered.</p> <ul style="list-style-type: none"> • In Data Quality, valid values are Logical Table and Physical Table. • In Stream Processing, the value is fixed to Task. • In API Service, the value is fixed to API.
Service	<p>The module where the alert event was triggered. Valid values:</p> <ul style="list-style-type: none"> • Data Quality • Stream Processing • API Service
Alert Cause	<p>The cause of the alert event. For more information, see View alert events.</p>
Push Method	<p>The method that Alert Center used to push alert messages for the alert event. Valid values:</p> <ul style="list-style-type: none"> • Cellphone • SMS • Email • DingTalk
Alert Frequency	<p>The frequency at which Alert Center pushed alert messages for the alert event.</p>
First Alerted At	<p>The time when Alert Center pushed the first alert message for the alert event.</p>
Last Alerted At	<p>The time when Alert Center pushed the last alert message for the alert event.</p>
Alert Recipients	<p>The recipients to whom Alert Center pushed alert messages for the alert event. For more information, see View alert events.</p>
Links	<p>The link to the module where the alert event was triggered and the link to the corresponding operational logs. The links that are provided for an alert event vary based on the module where the alert event may be triggered.</p> <ul style="list-style-type: none"> • Data Quality: The Go to Quality Check Rules and Go to Scheduling Task Instance Log links are provided. • Stream Processing: The Go to Stream Processing Task Monitoring Rules link is provided. • API Service: The Go to API Service Monitoring Rules link is provided.

Handle alert events

1. Go to the module where an alert event was triggered by using one of the following methods:
 - On the **All Events** tab, find the alert event and click the  icon in the **Actions** column.
 - On the **All Events** tab, click the name of the object that triggered the alert event. In the **Event Details** pane, click **Handle Alert**.
2. In the module where the alert event was triggered, handle the alert event based on the alert cause.

Change the status of alert events to Do-Not-Disturb

If alert events are in the **Alerting** or **Completed** state, you can change the status to **Do-Not-Disturb**.

1. On the **All Events** tab, open the **Alert Do-Not-Disturb Settings** dialog box by using one of the following methods:
 - Find an alert event and click the  icon in the **Actions** column.
 - Select multiple alert events in the **Alerting** or **Completed** state and click **Do-Not-Disturb** in the lower part of the page.
2. In the **Alert Do-Not-Disturb Settings** dialog box, set the **Duration** parameter to **All Day** and specify the **Specified Duration** parameter as required.
3. Click **OK**.

The status of one or more alert events changes from **Alerting** or **Completed** to **Do-Not-Disturb**.

9.6.4. Push records

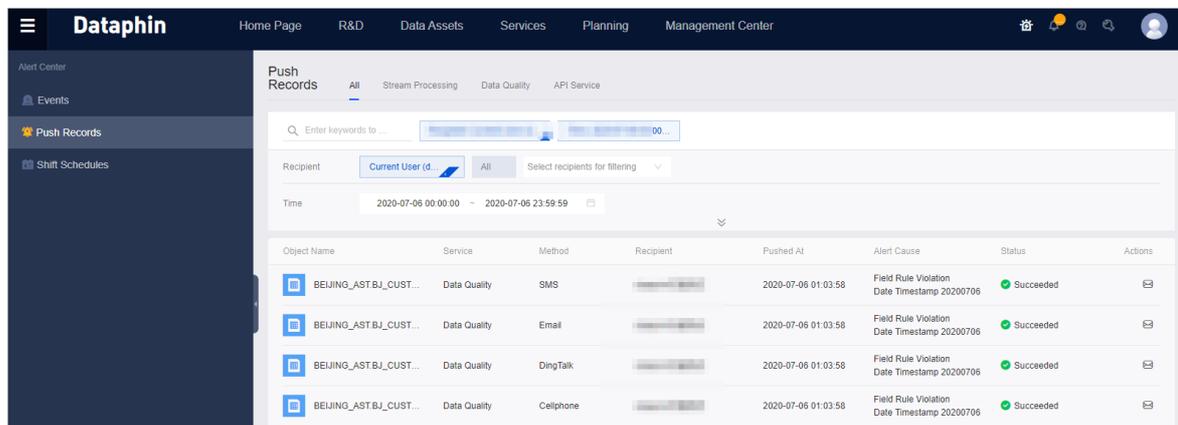
The Push Records page displays records about alert messages that Alert Center pushes. This topic describes how to view push records and the content of alert messages that are pushed.

Go to the Push Records page

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to **Alert Center**.
3. In **Alert Center**, click **Push Records** in the left-side navigation pane.

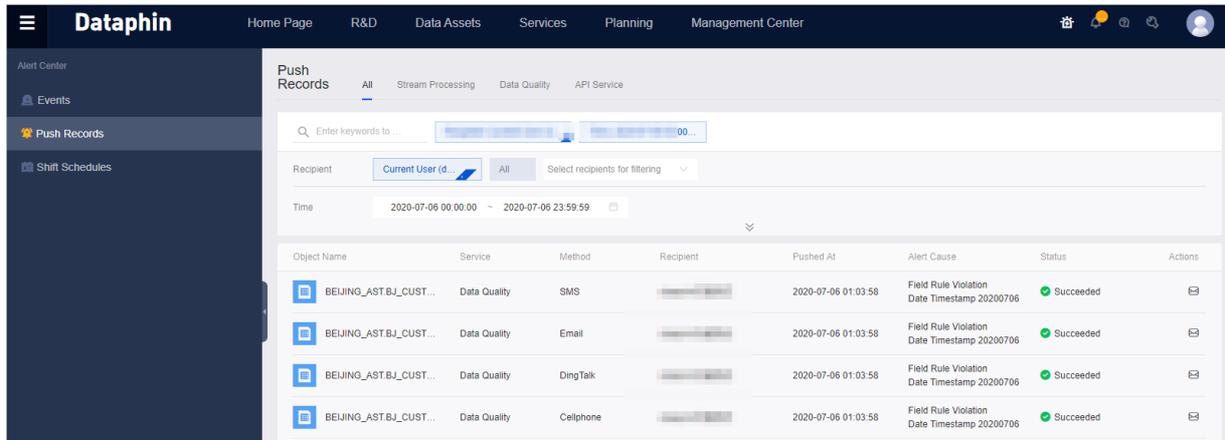
On the **Push Records** page, click the **Stream Processing**, **Data Quality**, or **API Service** tab to view the push records of the corresponding module.

To view all push records of the **Stream Processing**, **Data Quality**, and **API Service** modules, click the **All** tab. This topic describes push records on the **All** tab.



View push records

On the **All** tab, you can set the **Recipient**, **Time**, **Status**, and **Method** parameters to filter push records. You can also enter a keyword in the search box to search for push records.



The **All** tab displays the **Object Name**, **Service**, **Method**, **Recipient**, **Pushed At**, **Alert Cause**, and **Status** parameters of each push record. The **Actions** column of a push record lists the operations that you can perform on the push record.

Parameter	Description
Object Name	The name of the object that triggered the alert event.
Service	The module where the alert event was triggered. Valid values: <ul style="list-style-type: none"> • Data Quality • Stream Processing • API Service
Method	The method that Alert Center used to push the alert message to the specific recipients. Valid values: <ul style="list-style-type: none"> • Cellphone • SMS • Email • DingTalk
Recipient	The recipients of the alert message.
Pushed At	The time when the alert message was pushed.
Alert Cause	The cause of the alert event for which the alert message was pushed. The cause of an alert event varies based on the module where the alert event may be triggered. <ul style="list-style-type: none"> • In Data Quality, an alert event may be triggered due to the following causes: Table Rule Violation, Field Rule Violation, and Custom Rule Violation. • In API Service, an alert event may be triggered due to the following causes: Unexpected Average Response Time, Unexpected Number of Calls, Unexpected Error Rate, and Unexpected Percentage of Offline Tasks. • In Stream Processing, an alert event may be triggered due to the following causes: Excessively Long Delay, Exceeds Specified Data Retention Period, Exceeds Specified Failure Frequency, and Exceeds Maximum TPS.

Parameter	Description
Status	The pushing status of the alert message. Valid values: <ul style="list-style-type: none"> • Sending • Succeeded • Failed
Actions	To view the content of an alert message that was pushed, find the push record and click the  icon in the Actions column.

View the content of an alert message that was pushed

On the Push Records page, find a push record and click the  icon in the Actions column.

9.6.5. Shift schedules

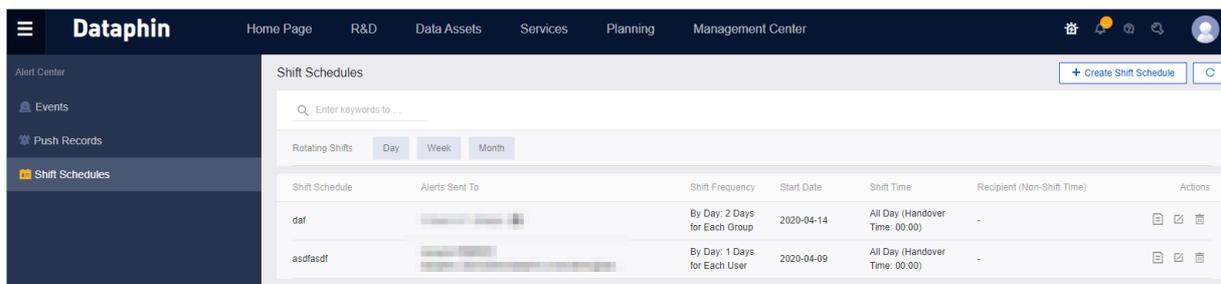
9.6.5.1. Overview

The Shift Schedules page displays shift schedules that are arranged for alert recipients who need to handle alerts.

To go to the Shift Schedules page, perform the following steps:

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to Alert Center.
3. In the left-side navigation pane, click Shift Schedules.

On the Shift Schedules page, you can set the Rotating Shifts parameter to filter shift schedules. You can also enter a keyword in the search box to search for shift schedules.



The Shift Schedules page displays the Shift Schedule, Alerts Sent To, Shift Frequency, Start Date, Shift Time, and Recipient (Non-Shift Time) parameters of each shift schedule. The Actions column of a shift schedule lists the operations that you can perform on the shift schedule.

Parameter	Description
Shift Schedule	The name of the shift schedule.

Parameter	Description
Alerts Sent To	<p>The recipients to whom Alert Center pushes alert messages during shifts that are arranged in the shift schedule.</p> <ul style="list-style-type: none"> If you set the Alerts Sent To parameter to Users when you create a shift schedule, the Alerts Sent To column of the shift schedule displays the specific user accounts. This column displays a maximum of 64 characters for each value. If a value contains more than 64 characters, excess characters are displayed as an ellipsis (...). If you set the Alerts Sent To parameter to Groups when you create a shift schedule, the Alerts Sent To column of the shift schedule displays the number of groups and total number of group members. You can move the pointer over the  icon to view group details.
Shift Frequency	The rotating frequency of shifts that are arranged in the shift schedule. For more information, see Create a shift schedule .
Start Date	The start date of shifts that are arranged in the shift schedule. For more information, see Create a shift schedule .
Shift Time	The duration of a shift. For more information, see Create a shift schedule .
Recipient (Non-Shift Time)	The recipient to whom Alert Center pushes alert messages in the non-shift duration of specific alert recipients. For example, the recipient can be the object owner or a specified user.
Actions	<p>The operations that you can perform on the shift schedule, including:</p> <ul style="list-style-type: none"> View details of the shift schedule. For more information, see View details of a shift schedule. Modify the shift schedule. For more information, see Modify a shift schedule. Delete the shift schedule. For more information, see Delete a shift schedule.

9.6.5.2. Create a shift schedule

The Shift Schedules page displays shift schedules that are arranged for alert recipients who need to handle alerts. This topic describes how to create a shift schedule.

Context

Only the super administrator and business unit administrators can create a shift schedule.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to **Alert Center**.
3. In the left-side navigation pane, click **Shift Schedules**.
4. On the **Shift Schedules** page, click **Create Shift Schedule** in the upper-right corner.
5. In the **Create Shift Schedule** dialog box, set the parameters as required.

Create Shift Schedule
✕

Configure Shift Schedule

* Name

* Alerts Sent To Users Groups

+ Add User

0/20

Shift Frequency By Day By Week By Month

Shift of Each User Day

* Start Date

Shift Time All Day Specified Time Range

Preview Shift Schedule

Section	Parameter	Description
	Name	The name of the shift schedule.
	Alerts Sent To	<p>The alert recipients for the shift schedule. Valid values: Users and Groups.</p> <ul style="list-style-type: none"> ◦ If you select Users, click Add User to add one or more users as alert recipients. <div style="border: 1px solid #add8e6; padding: 5px; margin: 5px 0;"> <p>? Note If the icon appears next to a user to be added as an alert recipient, you must add contact information for the user. For more information, see Manage members.</p> </div> <ul style="list-style-type: none"> ◦ If you select Groups, click Add User next to Group 1 and add alert recipients to the first group. If you want alert recipients in multiple groups to take turns, click Add Group to add more groups and add alert recipients to the groups. If an existing group is no longer needed, move the pointer over the group name and click the icon to delete the group.
Configure Shift Schedule		

Section	Parameter	Description
	Shift Frequency	<p>The rotating frequency of shifts that are arranged in the shift schedule. Valid values:</p> <ul style="list-style-type: none"> ◦ By Day: If you select this option, the value in the Shift of Each User or Shift of Each Group field next to this option cannot exceed 366, in units of days. ◦ By Week: If you select this option, the value in the Shift of Each User or Shift of Each Group field next to this option cannot exceed 53, in units of weeks. ◦ By Month: If you select this option, the value in the Shift of Each User or Shift of Each Group field next to this option cannot exceed 12, in units of months.
	Start Date	The start date of shifts that are arranged in the shift schedule.
	Shift Time	<p>The duration of a shift. Valid values: All Day and Specified Time Range.</p> <ul style="list-style-type: none"> ◦ If you select All Day, specify the Handover Time parameter. ◦ If you select Specified Time Range, specify a time range and the Recipient (Non-Shift Time) parameter. Valid values of the Recipient (Non-Shift Time) parameter are Object Owner and Specified User.
Preview Shift Schedule	Preview	You can click Preview to view the configured shift schedule. After you update the shift schedule, you can click Refresh Preview to update the preview data of the shift schedule.

6. Click OK.

9.6.5.3. View details of a shift schedule

The Shift Schedules page displays shift schedules that are arranged for alert recipients who need to handle alerts. This topic describes how to view details of a shift schedule.

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to **Alert Center**.
3. In the left-side navigation pane, click **Shift Schedules**.
4. On the **Shift Schedules** page, find the shift schedule whose details you want to view and click the  icon in the **Actions** column.

9.6.5.4. Modify a shift schedule

The Shift Schedules page displays shift schedules that are arranged for alert recipients who need to handle alerts. This topic describes how to modify a shift schedule.

Procedure

1. [Log on to the Dataphin console.](#)

2. On the Dataphin homepage, click the  icon in the upper-right corner to go to **Alert Center**.
3. In the left-side navigation pane, click **Shift Schedules**.
4. On the Shift Schedules page, find the shift schedule that you want to modify and click the  icon in the **Actions** column.
5. In the **Change Shift Schedule** dialog box, modify the parameters as needed. For more information, see [Create a shift schedule](#).
6. After you modify the parameters, click **OK**.

9.6.5.5. Delete a shift schedule

The Shift Schedules page displays shift schedules that are arranged for alert recipients who need to handle alerts. This topic describes how to delete a shift schedule.

Context

Limits:

- You cannot delete a shift schedule that is being referenced by the Stream Processing, Data Quality, or API Service module.
- Only the super administrator and business unit administrators can delete a shift schedule.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click the  icon in the upper-right corner to go to **Alert Center**.
3. In the left-side navigation pane, click **Shift Schedules**.
4. On the **Shift Schedules** page, find the shift schedule that you want to delete and click the  icon in the **Actions** column.
5. In the Tip message, click **OK**.

9.7. Data warehouse planning

9.7.1. Overview

Data warehouse planning is the first step in constructing a Data Mid-End based on Dataphin and an important step for the architectural design of your data system.

Background information

Data warehouse planning involves the architectural design of logical and physical spaces. The design helps you divide business domains and manage resources and projects.

- Logical spaces are namespaces and core objects that are defined based on business characteristics. Logical spaces include business units, data domains, and common business logic.
- Physical spaces are projects that are created to meet the requirements of development, collaboration, and management. Physical spaces support project management and data source management.

Go to the Planning page

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Planning** in the top navigation bar. You can also click **Intelligent Data**

Warehouse Planning in the middle section.

3. The **Planning** page consists of the **Business**, **Project**, and **Source** modules. In these modules, you can create data sources and computing engines, create projects based on data sources and computing engines, and bind projects to business units.

9.7.2. Business unit management

9.7.2.1. Create business units

As an important part of logical spaces, a business unit defines a data warehouse namespace in Dataphin based on business characteristics. A business unit can contain multiple projects. This topic describes how to create business units in Dev-Prod mode or a business unit in Basic mode.

Prerequisites

Dataphin members are added. For more information, see [Manage members](#).

Context

Only the Apsara Stack tenant account that is assigned the super administrator role can create business units.

Dataphin allows you to create business units in Dev-Prod or Basic mode. The following table describes the two modes.

Mode	Description
Dev-Prod mode	Generates a business unit in Dev mode as the development environment and a business unit in Prod mode as the production environment. The two business units are isolated. This allows you to guarantee data security in the production environment and manage the data production process and production data in a controllable way. We recommend that you select this mode if you are concerned more about management. This mode is applicable to the scenario where you have many developers who have clear responsibilities and you have a large budget for computing and storage.
Basic mode	Generates a business unit in Basic mode as an independent and flexible production environment. In this mode, the data production process is stable and easy-to-use and the production data is controllable. We recommend that you select this mode if you are concerned more about data development efficiency rather than management. This mode is applicable to the scenario where the responsibilities of developers are not clear and you have a limited budget for computing and storage.

Create business units in Dev-Prod mode

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, click **Create Business Unit** in the upper-right corner.
5. In the **Create Business Unit** dialog box, select Dev-Prod Mode, click **Next**, and then set the parameters as required.

Parameter	Description
Common Name	The common name of the business units. The common name can be up to 64 characters in length and can contain letters, digits, and underscores (_).
Common Display Name	<p>The common display name of the business units. The common display name can be up to 64 characters in length and can contain letters, digits, underscores (_), and hyphens (-).</p> <div style="background-color: #e1f5fe; padding: 5px;"> <p> Note By default, the name of the business unit in Dev mode is suffixed with _dev.</p> </div>
Description	The description of the business units.
Icon	<p>The icon that indicates the category of the business units. Valid values:</p> <ul style="list-style-type: none"> ◦  : E-Commerce ◦  : Finance ◦  : Cloud Computing ◦  : Advertising and Marketing ◦  : Logistics ◦  : Entertainment ◦  : Travel ◦  : Health ◦  : Socialization and Communications ◦  : Food & Drink ◦  : Education ◦  : Environment
Business Unit Administrators	The administrator of the business units. You can specify multiple administrators.

6. Click OK.

Create a business unit in Basic mode

1. On the **Business Units** page, click **Create Business Unit** in the upper-right corner.
2. In the **Create Business Unit** dialog box, select **Basic Mode**, click **Next**, and then set the parameters as required.

Parameter	Description
Name	The name of the business unit. The name can be up to 64 characters in length and can contain letters, digits, and underscores (_).
Display Name	The display name of the business unit. The display name can be up to 64 characters in length and can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the business unit.
Icon	<p>The icon that indicates the category of the business unit. Valid values:</p> <ul style="list-style-type: none"> ◦  : E-Commerce ◦  : Finance ◦  : Cloud Computing ◦  : Advertising and Marketing ◦  : Logistics ◦  : Entertainment ◦  : Travel ◦  : Health ◦  : Socialization and Communications ◦  : Food & Drink ◦  : Education ◦  : Environment
Business Unit Administrators	The administrator of the business unit. You can specify multiple administrators.

3. Click OK.

9.7.2.2. Modify a business unit

As an important part of logical spaces, a business unit defines a data warehouse namespace in Dataphin based on business characteristics. A business unit can contain multiple projects. This topic describes how to modify a business unit.

Prerequisites

Business units are created. For more information, see [Create business units](#).

Context

Only the Apsara Stack tenant account that is assigned the super administrator role can modify business units.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, move the pointer over the business unit that you want to modify and click the  icon.
5. In the **Change Business Unit** dialog box, change the parameter values as needed. For more information, see [Create business units](#).
6. Click OK.

9.7.2.3. Modify business unit parameters

Business unit parameters include the time-based partitioning parameter and the data timestamp parameter. This topic describes how to modify business unit parameters.

Prerequisites

Business units are created. For more information, see [Create business units](#).

Context

Only the Apsara Stack tenant account that is assigned the super administrator role can modify business unit parameters.

Business unit parameters function as common parameters of a business unit, such as the data type and default value. The following business unit parameters are supported:

- **Time-based Partitioning Field**: specifies the name, data type, default value, and description of the time-based partitioning parameter.
- **Data Timestamp**: specifies the name and format of the data timestamp parameter. By default, the data timestamp parameter is named bizdate.

Modify the time-based partitioning parameter

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:

- On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
 4. On the **Business Units** page, click the business unit for which you want to modify business unit parameters.
 5. In the **Business Unit Parameters** section, click the icon next to **Time-based Partitioning Field**.
 6. In the **Change Time Partition** dialog box, set the parameters as required.

Parameter	Description
Display Name	The display name of the time-based partitioning parameter. The display name can contain letters, digits, underscores (_), and hyphens (-).
Name	The name of the time-based partitioning parameter. Default value: ds . The name can contain letters, digits, and underscores (_).
Date Type	The data type of the time-based partitioning parameter. Valid values: <ul style="list-style-type: none"> ○ STRING ○ BIGINT ○ DOUBLE ○ DATETIME
Default Value	The default value of the time-based partitioning parameter. Enter NULL if you do not specify a default value.
Description	The description of the time-based partitioning parameter.

7. Click **OK**.

Modify the data timestamp parameter

1. On the **Business Units** page, click the business unit for which you want to modify business unit parameters.
2. In the **Business Unit Parameters** section, click the  icon next to **Data Timestamp**.
3. In the **Change Data Timestamp** dialog box, click the  icon next to **Parameter Value** and select a format for the data timestamp from the drop-down list.
4. Click **OK**.

9.7.2.4. Data domain management

9.7.2.4.1. Create a data domain

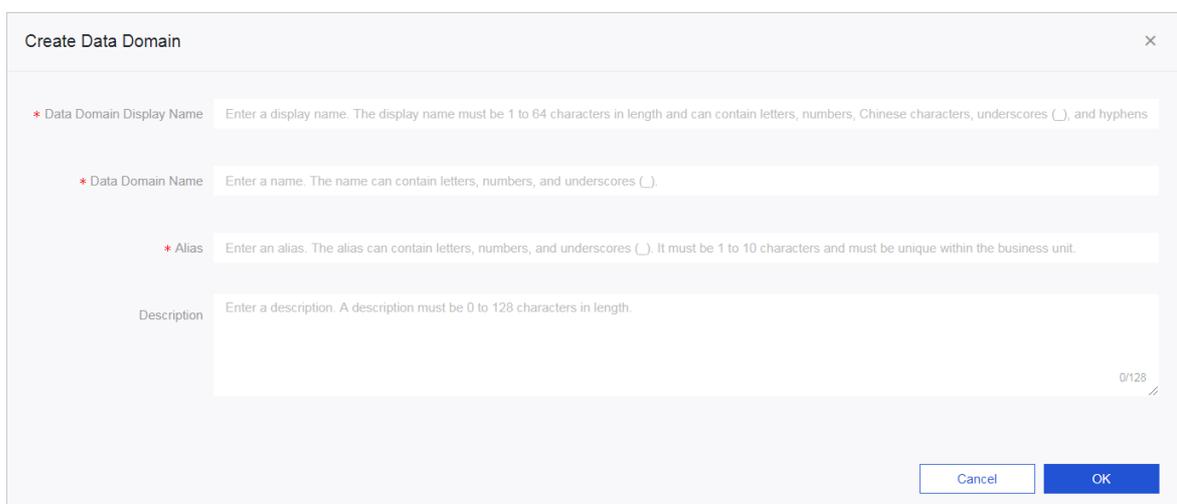
A data domain is used to categorize business concepts in a business unit, such as the commodity domain, transaction domain, and membership domain. This topic describes how to create a data domain.

Context

Only the super administrator and project administrators can create data domains.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, click the business unit for which you want to create a data domain.
5. In the **Data Domain** section, Click **Create Data Domain**.
6. In the **Create Data Domain** dialog box, set the parameters as required.



Parameter	Description
Data Domain Display Name	The display name of the data domain.
Data Domain Name	The name of the data domain.
Alias	The alias of the data domain. The alias cannot be the same as the name of the data domain.
Description	The description of the data domain.

7. Click **OK**.

9.7.2.4.2. Modify a data domain

A data domain is used to categorize business concepts in a business unit, such as the commodity domain, transaction domain, and membership domain. This topic describes how to modify a data domain.

Context

If a data domain is referenced, changing its name may result in errors in dependencies. In this case, we recommend that you do not modify the data domain name.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, click the business unit for which you want to modify a data domain.
5. In the **Data Domain** section, find the data domain that you want to modify and click the  icon in the **Actions** column.
6. In the **Change Data Domain** dialog box, change the parameter values as needed. For more information, see [Create a data domain](#).
7. Click **OK**.

9.7.2.4.3. Delete a data domain

A data domain is used to categorize business concepts in a business unit, such as the commodity domain, transaction domain, and membership domain. This topic describes how to delete a data domain.

Context

You cannot delete a data domain if it is referenced.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.

- On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, click the business unit for which you want to delete a data domain.
5. In the **Data Domain** section, find the data domain that you want to delete and click the  icon in the **Actions** column.
6. In the message that appears, click **OK**.

9.7.2.5. Delete a business unit

As an important part of logical spaces, a business unit defines a data warehouse namespace in Dataphin based on business characteristics. A business unit can contain multiple projects. This topic describes how to delete a business unit.

Prerequisites

Business units are created. For more information, see [Create business units](#).

Context

Limits:

- Only the Apsara Stack tenant account that is assigned the super administrator role can delete business units.
- To delete business units in Dev-Prod mode, make sure that the projects in the business units contain no tasks or objects.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Business Units** in the left-side navigation pane.
4. On the **Business Units** page, move the pointer over the business unit that you want to delete and click the  icon.
5. In the **Delete Business Unit** message, click **OK**.

9.7.3. Statistical period management

9.7.3.1. Create a statistical period

Dataphin supports only the common business logic of statistical periods. A statistical period indicates the time range for statistics, for example, the last seven days or the last 30 days. This topic describes how to create a statistical period.

Prerequisites

The computing engine type is set. For more information, see [Set the computing engine type](#). After the computing engine type is set, Dataphin automatically initializes the built-in statistical periods.

Context

Only the super administrator and project administrators can create statistical periods.

Procedure

1. Log on to the Dataphin console.
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Common Business Logic** in the left-side navigation pane.
4. On the **Common Business Logic** page, click **Create Statistical Period** in the upper-right corner.
5. In the **Create Statistical Period** dialog box, set the parameters as required.

Parameter	Description
Statistical Period	The display name of the statistical period.
Alias	The alias of the statistical period.
Description	The description of the statistical period.
Expression	The function expressions of the statistical period. You can click Expression Parameter Description next to Expression to view functions that Dataphin provides.
Start Time	The expression of the start time of the statistical period. You can set the expression of the start time by using one of the following methods: <ul style="list-style-type: none"> ○ Select Parameter and enter a parameter. ○ Select Function Expression, click the <input type="checkbox"/> icon, and then select a function expression.

Parameter	Description
End Time	<p>The expression of the end time of the statistical period. You can set the expression of the end time by using one of the following methods:</p> <ul style="list-style-type: none"> ◦ Select Parameter and enter a parameter. ◦ Select Function Expression, click the <input type="checkbox"/> icon, and then select a function expression.

 **Note** You cannot create a duplicate statistical period.

6. Click OK.

9.7.3.2. Modify a statistical period

A statistical period indicates the time range for statistics, for example, the last seven days or the last 30 days. This topic describes how to modify a statistical period.

Prerequisites

Statistical periods are created. For more information, see [Create a statistical period](#).

Context

Limits:

- The super administrator and project administrators can modify statistical periods.
- A project administrator cannot modify statistical periods that are created by other administrators. Members who are not administrators can modify only the statistical periods that are created by themselves.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Common Business Logic** in the left-side navigation pane.
4. On the **Common Business Logic** page, find the statistical period that you want to modify and click the icon in the **Actions** column.
5. In the **Change Statistical Period** dialog box, change the parameter values as needed. For more information, see [Create a statistical period](#).
6. Click OK.

9.7.3.3. Delete a statistical period

A statistical period indicates the time range for statistics, for example, the last seven days or the last 30 days. This topic describes how to delete a statistical period.

Prerequisites

Statistical periods are created. For more information, see [Create a statistical period](#).

Context

Limits:

- The super administrator and project administrators can delete statistical periods.
- A project administrator cannot delete statistical periods that are created by other administrators. Members who are not administrators can delete only the statistical periods that are created by themselves.
- If a published derived metric references a statistical period, such as the last 150 days, the statistical period cannot be deleted.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Common Business Logic** in the left-side navigation pane.
4. On the **Common Business Logic** page, find the statistical period that you want to delete and click the  icon in the **Actions** column.
5. In the message that appears, click **OK**.

9.7.4. Project management

9.7.4.1. Overview

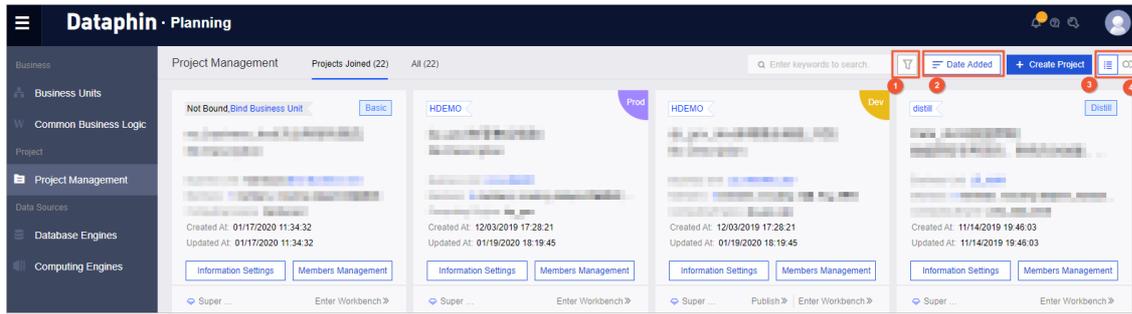
A project is used to isolate physical resources and developers during Data Mid-End construction. A business unit can contain multiple projects. Each Dataphin member can join multiple projects.

 **Note** Before you manage a project, make sure that the computing engine type of Dataphin is set on the Computing Engine Configuration page in Management Center. For more information, see [Set the computing engine type.](#)

On the **Project Management** page, you can find the projects that you join on the **Projects Joined** tab and all projects on the **All** tab. You can perform the following operations on the projects that you join:

- Configure project information, manage members, go to the **Publish** page, and go to the **Develop** page.
- In the **Information Settings** dialog box, configure the batch processing computing engine and stream processing computing engine for a project. For more information, see [Create a batch processing computing engine](#) and [Create a stream processing computing engine](#). Dataphin supports stream processing and batch processing. You can develop both stream processing tasks and batch processing tasks in the same project. This ensures consistent experience in development and improves the development efficiency.

You can also filter and sort projects and change the display mode of projects on the **Project Management** page, as described in the following table.



No.	Description
1	Click the  icon. In the More Filters dialog box, set the Business Units and Project Type parameters. Projects that meet the filter conditions appear.
2	Click the  icon and select Date Added , Display Name , or Name to display projects based on the dates when the projects are added in descending order or based on the display names or names in reverse alphabetical order.
3	Click the  icon. The projects are listed.
4	Click the  icon. The projects are categorized based on the project mode and projects in Dev-Prod mode are paired.

9.7.4.2. Create projects

A project is used to isolate physical resources and developers during Data Mid-End construction. This topic describes how to create projects in Dev-Prod mode or a project in Basic mode.

Prerequisites

Required computing engines are created. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).

Context

Limits:

- Only the super administrator and business unit administrators can create projects.
- A computing engine can be bound to only one project.
- You cannot change the computing engine of a project after they are bound.
- To make sure that tasks can be processed in a project in Prod mode, you must configure the stream processing computing engine or batch processing computing engine for both the project in Prod mode and the corresponding project in Dev mode.

Dataphin allows you to create projects in Dev-Prod or Basic mode. The following table describes the two modes.

Mode	Description
Dev-Prod mode	<p>Generates a project in Dev mode as the development environment and a project in Prod mode as the production environment. The two projects are isolated. This allows you to ensure data security in the production environment and manage the data production process in a controllable way. We recommend that you select this mode if you are concerned more about management. This mode is applicable to the scenario where you have many developers who have clear responsibilities and you have a large budget for computing and storage.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note By default, the name of the project in Dev mode is suffixed with <code>_dev</code>.</p> </div>
Basic mode	<p>Generates a project in Basic mode as an independent and flexible production environment. In this mode, the data production process is stable and easy-to-use and the production data is controllable. We recommend that you select this mode if you are concerned more about data development efficiency rather than management. This mode is applicable to the scenario where the responsibilities of developers are not clear and you have a limited budget for computing and storage.</p>

Create projects in Dev-Prod mode

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Project Management** in the left-side navigation pane.
4. On the **Project Management** page, click **Create Project** in the upper-right corner.
5. In the **Create Project** dialog box, select **Dev-Prod Mode**.
6. Click **Next**.
7. In the **Create Project** dialog box, set the parameters as required.

Section	Parameter	Description
	Common Display Name	<p>The common display name of the projects. Note the following naming rules when you set a name:</p> <ul style="list-style-type: none"> ◦ The common display name can contain letters, digits, underscores (<code>_</code>), and hyphens (<code>-</code>). ◦ By default, business unit names are prefixed with <code>LD_</code>. Therefore, project names cannot have this prefix. This enables Dataphin to distinguish between logical tables and physical tables in data queries. When logical tables and physical tables are referenced, the former is prefixed with the corresponding business unit name, and the latter is prefixed with the corresponding project name. If project names also start with <code>LD_</code>, Dataphin cannot distinguish between physical tables and logical tables that are both prefixed with <code>LD_</code>.

Name Settings Section	Parameter	Description
	Common Name	<p>The common name of the projects. Note the following naming rules when you set a name:</p> <ul style="list-style-type: none"> ◦ The common name can contain letters, digits, and underscores (_). ◦ If the computing engine type is MaxCompute, we recommend that you set Common Name to the MaxCompute project name. ◦ By default, business unit names are prefixed with LD_. Therefore, project names cannot have this prefix. This enables Dataphin to distinguish between logical tables and physical tables in data queries. When logical tables and physical tables are referenced, the former is prefixed with the corresponding business unit name, and the latter is prefixed with the corresponding project name. If project names also start with LD_, Dataphin cannot distinguish between physical tables and logical tables that are both prefixed with LD_.
Namespace	Business Unit	Optional. The business unit to which the project belongs.
	Project Type	<p>The type of the project. This parameter is used to categorize the tasks and generated data of projects. Default value: Application Data Store. Valid values:</p> <ul style="list-style-type: none"> ◦ Source Data: stores raw data of business databases. This layer serves as the source and basis in subsequent data development and is also called the vertical data center. ◦ Common Dimensional Modeling: extracts themes, standards, and common data from business data. This layer connects the source data layer and application data store layer and is also called the common data center. ◦ Application Data Store: defines diversified and distinct metrics based on business scenarios.
Project 1: Dev	Batch Processing	<p>The batch processing computing engine.</p> <ul style="list-style-type: none"> ◦ Make sure that the computing engine that you bind to each project is unique. Otherwise, a conflict may occur when data is written to the physical database. ◦ Batch processing computing engines are bound to MaxCompute projects. If you have multiple projects, each bound to a batch processing computing engine, make sure that your AccessKey pair has the permissions of the MaxCompute project administrator, including the permission to access data across the MaxCompute projects. If your AccessKey pair does not have the preceding permissions, you must execute the GRANT statement in the authorization code in MaxCompute to grant required permissions to your AccessKey pair. This ensures that you can pass the database authentication when you switch between projects in Dataphin. ◦ A batch processing computing engine can be bound to only one project.

Section	Parameter	Description
	Stream Processing	The stream processing computing engine.
	Name	Optional. The name of the project in Dev mode. Dataphin automatically generates a name based on the specified common name of the project in the format of Common name_dev.
	Display Name	Optional. The display name of the project in Dev mode. Dataphin automatically generates a name based on the specified common display name of the project in the format of Common display name_development.
	Description	The description of the project in Dev mode.
Project 2: Prod	Stream Processing	The stream processing computing engine.
	Batch Processing	The batch processing computing engine. For more information, see the description of the batch processing computing engine in Project 1: Dev.  Note You must bind different computing engines to the project in Dev mode and the project in Prod mode.
	Stream Processing	The stream processing computing engine.
	Name	Optional. The name of the project in Prod mode. Dataphin automatically generates a name based on the specified common name of the project.
	Display Name	Optional. The display name of the project in Prod mode. Dataphin automatically generates a name based on the specified common display name of the project.
	Description	The description of the project in Prod mode.
Other Settings	Sandbox Whitelist	The IP addresses, domain names, and database endpoints that Shell or Python tasks can access. To configure a sandbox whitelist, perform the following steps: <ul style="list-style-type: none"> i. Click Create. In the fields that appear, enter the IP address, domain name, or database endpoint and the port number. ii. Click the <input checked="" type="checkbox"/> icon to add the IP address, domain name, or database endpoint to the sandbox whitelist. <p>You can find an IP address, a domain name, or a database endpoint and click the  icon in the Actions column to remove it from the sandbox whitelist.</p>

8. Click OK.

Create a project in Basic mode

1. On the **Project Management** page, click **Create Project** in the upper-right corner.
2. In the **Create Project** dialog box, select **Basic Mode**.
3. Click **Next**.
4. In the **Create Project** dialog box, set the parameters as required.

Section	Parameter	Description
Basic Settings	Batch Processing	<p>The batch processing computing engine.</p> <ul style="list-style-type: none"> ◦ Make sure that the computing engine that you bind to each project is unique. Otherwise, a conflict may occur when data is written to the physical database. ◦ Batch processing computing engines are bound to MaxCompute projects. If you have multiple projects, each bound to a batch processing computing engine, make sure that your AccessKey pair has the permissions of the MaxCompute project administrator, including the permission to access data across the MaxCompute projects. If your AccessKey pair does not have the preceding permissions, you must execute the GRANT statement in the authorization code in MaxCompute to grant required permissions to your AccessKey pair. This ensures that you can pass the database authentication when you switch between projects in Dataphin. ◦ A batch processing computing engine can be bound to only one project.
	Stream Processing	The stream processing computing engine.
	Name	<p>The name of the project. Note the following naming rules when you set a name:</p> <ul style="list-style-type: none"> ◦ The name can contain letters, digits, and underscores (_). ◦ By default, business unit names are prefixed with LD_. Therefore, project names cannot have this prefix. This enables Dataphin to distinguish between logical tables and physical tables in data queries. When logical tables and physical tables are referenced, the former is prefixed with the corresponding business unit name, and the latter is prefixed with the corresponding project name. If project names also start with LD_, Dataphin cannot distinguish between physical tables and logical tables that are both prefixed with LD_.

Section	Parameter	Description
	Display Name	<p>The display name of the project. Note the following naming rules when you set a name:</p> <ul style="list-style-type: none"> ◦ The display name can contain letters, digits, underscores (_), and hyphens (-). ◦ By default, business unit names are prefixed with LD_. Therefore, project names cannot have this prefix. This enables Dataphin to distinguish between logical tables and physical tables in data queries. When logical tables and physical tables are referenced, the former is prefixed with the corresponding business unit name, and the latter is prefixed with the corresponding project name. If project names also start with LD_, Dataphin cannot distinguish between physical tables and logical tables that are both prefixed with LD_.
	Description	The description of the project in Basic mode.
Namespace	Business Unit	Optional. The business unit to which the project belongs.
	Project Type	<p>The type of the project. This parameter is used to categorize the tasks and generated data of projects. Default value: Application Data Store. Valid values:</p> <ul style="list-style-type: none"> ◦ Source Data: stores raw data of business databases. This layer serves as the source and basis in subsequent data development and is also called the vertical data center. ◦ Common Dimensional Modeling: extracts themes, standards, and common data from business data. This layer connects the source data layer and application data store layer and is also called the common data center. ◦ Application Data Store: defines diversified and distinct metrics based on business scenarios.
Other Settings	Sandbox Whitelist	<p>The IP addresses, domain names, and database endpoints that Shell or Python tasks can access. To configure a sandbox whitelist, perform the following steps:</p> <ol style="list-style-type: none"> Click Create. In the fields that appear, enter the IP address, domain name, or database endpoint and the port number. Click the <input checked="" type="checkbox"/> icon to add the IP address, domain name, or database endpoint to the sandbox whitelist. <p>You can find an IP address, a domain name, or a database endpoint and click the  icon in the Actions column to remove it from the sandbox whitelist.</p>

5. Click OK.

9.7.4.3. Configure projects

You can configure project information and manage project members. This topic describes how to configure projects in Dev-Prod mode or a project in Basic mode.

Prerequisites

A project in Basic mode is created or projects in Dev-Prod mode are created. For more information, see [Create projects](#).

Context

Limits:

- By default, the super administrator also plays the administrator role in all projects. You cannot delete or change the super administrator role.
- Only project administrators can change the roles of other project members.
- Only the super administrator and project administrators can configure project information.

Configure the information about projects in Dev-Prod mode

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Project Management** in the left-side navigation pane.
4. On the Project Management page, find the project that you want to configure and click **Information Settings**.
5. In the **Information Settings** dialog box, change the parameter values as needed and set the parameters in the **General Feature Control** section. For more information, see [Create projects](#).
 - If the **Data Result Download** parameter is set to **Yes**, the roles except the guests can download result data.
 - If the **Data Result Download** parameter is set to **No**, the result data cannot be downloaded and can only be viewed on the Result tab.
6. Click **OK**.

Configure the information about a project in Basic mode

1. On the **Project Management** page, find the project that you want to configure and click **Information Settings**.
2. In the **Information Settings** dialog box, change the parameter values as needed and set the parameters in the **General Feature Control** section. For more information, see [Create projects](#).
 - If the **Data Result Download** parameter is set to **Yes**, the roles except the guests can download result data.
 - If the **Data Result Download** parameter is set to **No**, the result data cannot be downloaded and can only be viewed on the Result tab.
3. Click **OK**.

Add members

1. On the **Project Management** page, find the project that you want to configure and click **Members Management**.
2. In the **Manage Members** dialog box, click **Add Members**.
3. Add one or more members.

- i. In the **Add Members** dialog box, set the parameters as required.

Parameter	Description
Username	The user to be added as a project member. You can specify multiple users.
Role	The role that you want the selected users to assume.

- ii. Click **OK**.
4. After you set the preceding parameters, click **OK**.

Change the role of a member

1. On the **Project Management** page, find the project that you want to configure and click **Members Management**.
2. In the **Manage Members** dialog box, find the member whose role you want to change and click the  icon in the **Role** column.
3. From the drop-down list, select a role.
4. Click **OK**.

Remove a member

1. On the **Project Management** page, find the project that you want to configure and click **Members Management**.
2. Remove a member.
 - i. In the **Manage Members** dialog box, find the member you want to remove and click the  icon in the **Actions** column.
 - ii. In the message that appears, click **OK**.
3. In the **Manage Members** dialog box, click **OK**.

Remove multiple members at a time

1. On the **Project Management** page, find the project that you want to configure and click **Members Management**.
2. Remove multiple members at a time.
 - i. In the **Manage Members** dialog box, select multiple members to be removed.
 - ii. Click **Remove** in the lower part of the **Manage Members** dialog box.
 - iii. In the message that appears, click **OK**.
3. In the **Manage Members** dialog box, click **OK**.

9.7.4.4. Delete projects

A project is used to isolate physical resources and developers during Data Mid-End construction. This topic describes how to delete projects.

Prerequisites

A project in Basic mode is created or projects in Dev-Prod mode are created. For more information, see [Create projects](#).

Context

Limits:

- Only the super administrator and project administrators can delete projects.
- Only projects that contain no tasks or objects, such as code tasks or data standardization and data modeling objects, can be deleted.
- To delete projects in Dev-Prod mode, make sure that the project in Dev mode and the project in Prod mode contain no tasks or objects.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the Planning page, click **Project Management** in the left-side navigation pane.
4. In the lower part of the **Project Management** page, click **Enable Batch Deletion**.
5. Select the projects to be deleted. To delete projects in Dev-Prod mode, you can select only the project in Dev mode or the project in Prod mode. The other project is automatically selected.
6. In the lower part of the page, click **Delete**.
7. In the message that appears, click **OK**.

9.7.5. Maintenance and upgrade

9.7.5.1. Important notes for upgrade

This topic describes the important notes for upgrading business units and projects.

If you want to upgrade Dataphin or control Dataphin for data security purposes, you can lock Dataphin by setting Dataphin to work in the maintenance mode.

When the business grows or the publishing process of some data must be controlled, you can upgrade business units or projects from the Basic mode to the Dev-Prod mode. During maintenance and upgrade, data read and write operations are not allowed, but tasks run normally.

Important notes for upgrading business units

- Only the super administrator can upgrade business units.
- You cannot downgrade business units after they are upgraded.
- You cannot upgrade business units whose names are suffixed with `_dev`.
- You can upgrade only one business unit at a time.
- If you upgrade a business unit in the Basic mode that has no projects, the business unit is upgraded to a business unit in the Prod mode. A corresponding business unit in the Dev mode is created. The two business units share all configurations and resources.
- If you upgrade a business unit in the Basic mode that has projects with data standardization and data modeling objects, the projects are forcibly upgraded to projects in the Prod mode. Corresponding projects in the Dev mode are created.
- If you upgrade a business unit in the Basic mode that has projects without data standardization or data modeling objects, the projects remain in the Basic mode. However, the data standardization and data modeling features of these projects are disabled during and after the upgrade. For example, when you open a logical aggregate table found by global search in these projects, you can only view its information, but cannot perform any operations.

Important notes for upgrading projects

- Only the super administrator can upgrade projects.
- You cannot downgrade projects after they are upgraded.
- You can upgrade one or more projects at a time.
- If you upgrade a project in the Basic mode that is bound to a business unit, the project is upgraded to a project in the Prod mode. A corresponding project in the Dev mode is created. The data standardization and data modeling features are enabled for both projects after the upgrade.
- If you upgrade a project in the Basic mode, Dataphin upgrades the project to a project in the Prod mode and copies the configuration of the original project in the Basic mode to create a project in the Dev mode. Objects in the draft state in the original project are automatically deleted.
- After a project in the Basic mode is upgraded, you can publish objects in the project in the Dev mode to overwrite the corresponding objects in the developing state in the project in the Prod mode.

9.7.5.2. Maintain and upgrade business units

This topic describes how to maintain and upgrade business units.

Maintain business units

1. [Log on to the Dataphin console](#).
2. Open the **Maintenance and Upgrade** dialog box by using one of the following methods:
 - a. On the Dataphin homepage, click **Planning** in the top navigation bar.
You can also click **Intelligent Data Warehouse Planning** in the middle section.
 - b. On the **Planning** page, click **Business Units** in the left-side navigation pane.
 - c. On the **Business Units** page, click **Maintenance and Upgrade** in the upper-right corner.
 - On the Dataphin homepage, move the pointer over the  icon in the upper-right corner and select **Start Maintenance and Upgrade**.
3. In the **Maintenance and Upgrade** dialog box, set the **Choose a type** parameter to **Maintain**. Click **Next**.
4. In the Maintenance Configuration step, set the **Start Time**, **Phone Number**, and **Contact Email Address** parameters, select **I know the impact of this operation and confirm that I need to start the maintenance mode within the tenant to maintain or upgrade the system**, and then click **Start**.

If you want to cancel the maintenance before the maintenance starts, perform the following steps:

- i. In the **Maintenance and Upgrade** dialog box, click **Cancel Maintenance**.
- ii. In the message that appears, click **OK**.

After the maintenance starts, you cannot perform operations on modules. If you want to stop the ongoing maintenance, perform the following steps as the super administrator:

- i. Go to the **Starting Maintenance** page and click **Forcedly Stop Maintenance**.
- ii. In the message that appears, click **OK**.

Upgrade business units

1. In the **Maintenance and Upgrade** dialog box, set the **Choose a type** parameter to **Upgrade to Dev-Prod Mode**. Click **Next**.
2. In the **Maintenance Configuration** step, set the **Start Time**, **Phone Number**, and **Contact Email Address** parameters, select **I know the impact of this operation and confirm that I need to start the maintenance mode within the tenant to maintain or upgrade the system**, and then click **Start**.

If you want to cancel the upgrade before the upgrade starts, perform the following steps:

- i. In the **Maintenance and Upgrade** dialog box, click **Cancel Maintenance**.
- ii. In the message that appears, click **OK**.

After the upgrade starts, you cannot perform operations on modules. If you want to stop the ongoing upgrade, perform the following steps as the super administrator:

- i. On the left of the **Upgrade Configuration** page, click **Stop Maintenance**.
- ii. In the message that appears, click **OK**.

3. After the upgrade is completed, click **Upgraded**.
4. On the page that appears, click **Continue to Upgrade** or **Exit Maintenance Mode**.

9.7.5.3. View maintenance and upgrade logs

This topic describes how to view maintenance and upgrade logs.

Procedure

1. Log on to the Dataphin console.
2. On the Dataphin homepage, move the pointer over the  icon in the upper-right corner and select **View History**.
3. In the **Operations Log for Maintenance and Upgrades** dialog box, view the operators, operation time, and operations.
4. Click **OK**.

9.7.6. Data source

9.7.6.1. Overview

Data source overview

A data source is a physical data store. You can create data sources to import business data to Dataphin, based on which you can build a Data Mid-End. You can also import existing data to a data source.

Data source types

The following table describes the types of data sources that Dataphin supports.

Category	Data source type
Batch processing	MaxCompute, MySQL, SQL Server, PostgreSQL, Oracle, Hadoop Distributed File System (HDFS), Hive, FTP, Vertica, PolarDB-X, AnalyticDB, Elasticsearch, HBase_1_1_X, HBase_0_9_4, MongoDB, AnalyticDB for MySQL V3.0, AnalyticDB for PostgreSQL, and LogHub
Stream processing	PolarDB-X, DataHub, ApsaraDB for HBase, Log Service, Tablestore, Kafka_9_11, and RocketMQ

Access methods

Dataphin can use different methods to connect to different types of databases.

- Non-ApsaraDB databases accessible from the Internet: After Dataphin is granted the access to a non-ApsaraDB database accessible from the Internet, you can connect Dataphin to the database by using the public IP address or endpoint of the database.
- ApsaraDB databases: If an ApsaraDB database is accessible from the Internet, you can connect Dataphin to the database by using the public IP address of the database after Dataphin is granted the access to the database. If an ApsaraDB database is in a virtual private cloud (VPC), you can connect Dataphin to the database by using the VPC after the IP address of Dataphin is added to the whitelist of the database.
- User-created databases on ECS instances in VPCs: After the IP address of Dataphin is added to the whitelist of a user-created database on an ECS instance in a VPC, you can connect Dataphin to the database by using the VPC.
- Databases in a MaxCompute project: If a database is used in a MaxCompute project, you can connect Dataphin to the database by using the endpoint of the database.

Permissions

- All Dataphin members can create data sources.
- The super administrator can delete and modify all data sources, test the connectivity of all data sources, and change the owners of all data sources.
- Members other than the super administrator can only delete and modify their own data sources, test the

connectivity of their own data sources, and change the owner of their own data sources.

9.7.6.2. Create batch processing data sources

9.7.6.2.1. Create a MaxCompute data source

This topic describes how to create a MaxCompute data source.

Prerequisites

- The endpoint that you can use to connect to MaxCompute is obtained.
- The name of a MaxCompute project is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin connects to the MaxCompute project are obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
 - The **Data Sources** page displays the information about all created data sources, including the name, owner, supported data processing type, creation information, and connection information of each data source.
 - In the upper-right corner of the **Data Sources** page, you can enter a keyword in the search box to search for a specific data source by name.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select MAX_COMPUTE .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.

Parameter	Description
Endpoint	The endpoint that you can use to connect to the MaxCompute project.
Project Name	The name of the MaxCompute project to which the data source is connected.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the MaxCompute project.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the MaxCompute project.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.2. Create a MySQL data source

This topic describes how to create a MySQL data source.

Prerequisites

- The JDBC URL that you can use to connect to a MySQL database is obtained.
- The username and password that you can use to connect to the MySQL database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the Planning page, click **Data Sources** in the left-side navigation pane.
 - The **Data Sources** page displays the information about all created data sources, including the name, owner, supported data processing type, creation information, and connection information of each data source.
 - In the upper-right corner of the **Data Sources** page, you can enter a keyword in the search box to search for a specific data source by name.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select MYSQL .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the MySQL database.
Username	The username that you can use to connect to the MySQL database.
Password	The password that you can use to connect to the MySQL database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.3. Create an SQL Server data source

This topic describes how to create an SQL Server data source.

Prerequisites

- The JDBC URL that you can use to connect to an SQL Server database is obtained.
- The schema of the SQL Server database is obtained.
- The username and password that you can use to connect to the SQL Server database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select SQL_SERVER .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the SQL Server database.
Schema	The schema of the SQL Server database.
Username	The username that you can use to connect to the SQL Server database.
Password	The password that you can use to connect to the SQL Server database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.4. Create a PostgreSQL data source

This topic describes how to create a PostgreSQL data source.

Prerequisites

- The JDBC URL that you can use to connect to a PostgreSQL database is obtained.
- The schema of the PostgreSQL database is obtained.
- The username and password that you can use to connect to the PostgreSQL database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select POSTGRE_SQL .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).

Parameter	Description
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the PostgreSQL database.
Schema	The schema of the PostgreSQL database.
Username	The username that you can use to connect to the PostgreSQL database.
Password	The password that you can use to connect to the PostgreSQL database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.5. Create an Oracle data source

This topic describes how to create an Oracle data source.

Prerequisites

- The JDBC URL that you can use to connect to an Oracle database is obtained.
- The schema of the Oracle database is obtained.
- The username and password that you can use to connect to the Oracle database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ORACLE .

Parameter	Description
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the Oracle database.
Schema	The schema of the Oracle database.
Username	The username that you can use to connect to the Oracle database.
Password	The password that you can use to connect to the Oracle database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.6. Create an HDFS data source

This topic describes how to create an HDFS data source.

Prerequisites

- The JDBC URL that you can use to connect to HDFS is obtained.
- The URL that you can use to connect to an HDFS file system is obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select HDFS .

Parameter	Description
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
DefaultFS	The URL of the HDFS file system, in the format of <code>hdfs://Server IP:Port</code> .
Kerberos	<p>Specifies whether to enable Kerberos authentication. You can select Enable or Disable. If you select Enable, you must also set the following parameters:</p> <ul style="list-style-type: none"> ◦ KDC Server: the endpoint of the key distribution center (KDC) server. Separate multiple endpoints with commas (,). ◦ Keytab File: the keytab file for Kerberos authentication. ◦ Principal: the Kerberos principal name for Kerberos authentication.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.7. Create an FTP data source

This topic describes how to create an FTP data source.

Prerequisites

- The IP address of an FTP server is obtained.
- The port number of the FTP server is obtained.
- The username and password that you can use to connect to the FTP server are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select FTP .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
Protocol	<p>The protocol that is used to transfer files. You can select FTP or SFTP.</p> <ul style="list-style-type: none"> ○ FTP: transfers files by using FTP. ○ SFTP: transfers files by using SFTP.
Host	The name of the FTP server.
Port	The port number of the FTP server.
Username	The username that you can use to connect to the FTP server.
Password	The password that you can use to connect to the FTP server.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.8. Create a Hive data source

This topic describes how to create a Hive data source.

Prerequisites

- The following cluster configurations of Hive are obtained:
 - The hostname of the Hive cluster
 - The IP address of the Hive cluster
 - The port number of the Hive cluster
- The following HDFS configurations of Hive are obtained:
 - The keytab file for Kerberos authentication for HDFS
 - The Kerberos principal name for Kerberos authentication for HDFS
- The following Hive configurations are obtained:
 - The JDBC URL that you can use to connect to Hive

- The keytab file and the Kerberos principal name for Kerberos authentication for Hive
- The following configurations of the metadata base are obtained:
 - **JDBC URL**
 - The username that you can use to connect to the metadata base
 - The password that you can use to connect to the metadata base

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Section	Parameter	Description
Basic information	Type	The type of the data source. Select HIVE .
	Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
	Description	The description of the data source.
	Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
Cluster Configuration	NameNode	The location of the current file system root, in the format of Hostname:Port or IP address:Port. Specify the hostname, port number, and IP address of the Hive cluster.
	Kerberos	Specifies whether to enable Kerberos authentication. You can select Enable or Disable . If you select Enable , you must set the KDC Server parameter or upload the Kerberos configuration file. The KDC Server parameter specifies the endpoint of the KDC server. Separate multiple endpoints with commas (,).
	KRB5 File	The Kerberos configuration file that is used for authentication.
	HDFS Keytab File	The keytab file for Kerberos authentication for HDFS.

Section Configuration	Parameter	Description
	HDFS Principal	The Kerberos principal name for Kerberos authentication for HDFS.
Hive Configuration	JDBC URL	The JDBC URL that you can use to connect to Hive.
	Hive Keytab File	The keytab file for Kerberos authentication for Hive.
	Hive Principal	The Kerberos principal name for Kerberos authentication for Hive.
Metadatabase Configuration	Database Type	The type of the metadatabase. Valid values: MySQL and PostgreSQL .
	JDBC URL	The JDBC URL that you can use to connect to the metadatabase.
	Username	The username that you can use to connect to the metadatabase.
	Password	The password that you can use to connect to the metadatabase.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.9. Create an Elasticsearch data source

This topic describes how to create an Elasticsearch data source.

Prerequisites

- The URL that you can use to connect to Elasticsearch is obtained.
- The username and password that you can use to connect to Elasticsearch are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ELASTIC_SEARCH .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
ES URL	The URL that you can use to connect to Elasticsearch.
Username	The username that you can use to connect to Elasticsearch.
Password	The password that you can use to connect to Elasticsearch.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.10. Create a MongoDB data source

This topic describes how to create a MongoDB data source.

Prerequisites

- The JDBC URL that you can use to connect to a MongoDB database is obtained.
- The username and password that you can use to connect to the MongoDB database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select MONGODB .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the MongoDB database.
Username	The username that you can use to connect to the MongoDB database.
Password	The password that you can use to connect to the MongoDB database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.11. Create an AnalyticDB data source

This topic describes how to create an AnalyticDB data source.

Prerequisites

- The JDBC URL that you can use to connect to an AnalyticDB database is obtained.
- The username and password that you can use to connect to the AnalyticDB database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ANALYTICDB .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the AnalyticDB database.
Username	The username that you can use to connect to the AnalyticDB database.
Password	The password that you can use to connect to the AnalyticDB database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.12. Create a PolarDB-X data source

This topic describes how to create a PolarDB-X data source.

Prerequisites

- The JDBC URL that you can use to connect to a PolarDB-X database is obtained.
- The username and password that you can use to connect to the PolarDB-X database are obtained.

Procedure

1. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
2. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
3. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
4. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select DRDS .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.

Parameter	Description
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the PolarDB-X database.
Username	The username that you can use to connect to the PolarDB-X database.
Password	The password that you can use to connect to the PolarDB-X database.

5. Click **Test Connection** to test the connectivity of the data source.
6. After the data source passes the connectivity test, click **OK**.

9.7.6.2.13. Create a Vertica data source

This topic describes how to create a Vertica data source.

Prerequisites

- The JDBC URL that you can use to connect to a Vertica database is obtained.
- The schema of the Vertica database is obtained.
- The username and password that you can use to connect to the Vertica database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
 - The **Data Sources** page displays the information about all created data sources, including the name, owner, supported data processing type, creation information, and connection information of each data source.
 - In the upper-right corner of the **Data Sources** page, you can enter a keyword in the search box to search for a specific data source by name.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select VERTICA .

Parameter	Description
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ◦ If you want to use the data source only in the production environment, select Prod. ◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the Vertica database.
Schema	The schema of the Vertica database.
Username	The username that you can use to connect to the Vertica database.
Password	The password that you can use to connect to the Vertica database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.14. Create an HBase data source

This topic describes how to create an HBase data source.

Prerequisites

- The endpoints of KDC servers are obtained.
- The Kerberos principal name for Kerberos authentication is obtained.
- The keytab file for Kerberos authentication is obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select HBASE_0_9_4 or HBASE_1_1_X .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
Endpoint	The endpoint that you can use to connect to an HBase database.
Connection Parameters	The parameters that are required to connect to the HBase database. Enter the parameters in the JSON format.
Kerberos	<p>Specifies whether to enable Kerberos authentication. You can select Enable or Disable. If you select Enable, you must also set the following parameters:</p> <ul style="list-style-type: none"> ○ KDC Server: the endpoint of the KDC server. Separate multiple endpoints with commas (,). ○ Keytab File: the keytab file for Kerberos authentication. ○ Principal: the Kerberos principal name for Kerberos authentication.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.15. Create an AnalyticDB for MySQL V3.0 data source

This topic describes how to create an AnalyticDB for MySQL V3.0 data source.

Prerequisites

- The JDBC URL that you can use to connect to an AnalyticDB for MySQL V3.0 database is obtained.
- The username and password that you can use to connect to the AnalyticDB for MySQL V3.0 database are obtained.
- The ID of the cluster where the AnalyticDB for MySQL V3.0 database resides is obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.

3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ADB_FOR_MYSQL_V3 .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the AnalyticDB for MySQL V3.0 database.
Username	The username that you can use to connect to the AnalyticDB for MySQL V3.0 database.
Password	The password that you can use to connect to the AnalyticDB for MySQL V3.0 database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.16. Create an AnalyticDB for PostgreSQL data source

This topic describes how to create an AnalyticDB for PostgreSQL data source.

Prerequisites

- The JDBC URL that you can use to connect to an AnalyticDB for PostgreSQL database is obtained.
- The schema of the AnalyticDB for PostgreSQL database is obtained.
- The username and password that you can use to connect to the AnalyticDB for PostgreSQL database are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.

3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ADB_FOR_PG .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none">◦ If you want to use the data source only in the production environment, select Prod.◦ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods:<ul style="list-style-type: none">▪ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment.▪ Select Prod and Dev and configure the data source for both the production and development environments.
JDBC URL	The JDBC URL that you can use to connect to the AnalyticDB for PostgreSQL database.
Schema	The schema of the AnalyticDB for PostgreSQL database.
Username	The username that you can use to connect to the AnalyticDB for PostgreSQL database.
Password	The password that you can use to connect to the AnalyticDB for PostgreSQL database.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.2.17. Create a LogHub data source

This topic describes how to create a LogHub data source.

Prerequisites

- The endpoint that you can use to connect to LogHub is obtained.
- The name of a LogHub project is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin connects to the LogHub project are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:

- On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
 4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
 5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select LOG_HUB .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
LogHub Endpoint	The endpoint that you can use to connect to LogHub.
Project	The name of the LogHub project.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the LogHub project.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the LogHub project.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3. Create stream processing data sources

9.7.6.3.1. Create a DataHub data source

This topic describes how to create a DataHub data source.

Prerequisites

- The endpoint that you can use to connect to DataHub is obtained.
- The name of a DataHub project is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin connects to the DataHub project are obtained.

Procedure

1. Log on to the Dataphin console.
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select DATAHUB .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none">○ If you want to use the data source only in the production environment, select Prod.○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods:<ul style="list-style-type: none">■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment.■ Select Prod and Dev and configure the data source for both the production and development environments.
Endpoint	The endpoint that you can use to connect to DataHub.
Project Name	The name of the DataHub project.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the DataHub project.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the DataHub project.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3.2. Create a Log Service data source

This topic describes how to create a Log Service data source.

Prerequisites

- The endpoint that you can use to connect to Log Service is obtained.
- The name of a Log Service project is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin

connects to the Log Service project are obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select LOG_SERVICE .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
Endpoint	The endpoint that you can use to connect to Log Service.
Project Name	The name of the Log Service project.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the Log Service project.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the Log Service project.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3.3. Create an ApsaraDB for HBase data source

This topic describes how to create an ApsaraDB for HBase data source.

Prerequisites

- The endpoint that you can use to connect to the quorum servers of ZooKeeper built in ApsaraDB for HBase is obtained.

- The endpoint that you can use to connect to an ApsaraDB for HBase database is obtained.
- The username and password that you can use to connect to the ApsaraDB for HBase database are obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ALIYUN_HBASE .
Name	The name of the data source.
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none">○ If you want to use the data source only in the production environment, select Prod.○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods:<ul style="list-style-type: none">■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment.■ Select Prod and Dev and configure the data source for both the production and development environments.
Service type	The edition of ApsaraDB for HBase. Valid values: STANDARD and ENHANCED . <ul style="list-style-type: none">○ If you set the Service type parameter to STANDARD, you must also set the Version and zkQuorum parameters.○ If you set the Service type parameter to ENHANCED, you must also set the endPoint, Username, and Password parameters.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3.4. Create a Kafka data source

This topic describes how to create a Kafka data source.

Prerequisites

The endpoint that you can use to connect to a Kafka cluster is obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select KAFKA_9_11 .
Name	The name of the data source.
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
Kafka cluster address	The endpoint that you can use to connect to the Kafka cluster.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3.5. Create a Tablestore data source

This topic describes how to create a Tablestore data source.

Prerequisites

- The endpoint that you can use to connect to a Tablestore instance is obtained.
- The name of the Tablestore instance is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin connects to the Tablestore instance are obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.

4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select TABLE_STORE .
Name	The name of the data source.
Description	The description of the data source.
Environment	The type of the environment where the data source will be used. <ul style="list-style-type: none">○ If you want to use the data source only in the production environment, select Prod.○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods:<ul style="list-style-type: none">■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment.■ Select Prod and Dev and configure the data source for both the production and development environments.
Endpoint	The endpoint that you can use to connect to the Tablestore instance.
Access Id	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the Tablestore instance.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the Tablestore instance.
Instance Name	The name of the Tablestore instance.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.3.6. Create a RocketMQ data source

This topic describes how to create a RocketMQ data source.

Prerequisites

- The endpoint that you can use to connect to a RocketMQ instance is obtained.
- The ID of the RocketMQ instance is obtained.
- The AccessKey ID and AccessKey secret that are used to authenticate the user identity when Dataphin connects to the RocketMQ instance are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.

3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the **Data Sources** page, click **Create Data Source** in the upper-right corner.
5. In the **Create Data Source** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the data source. Select ROCKET_MQ .
Name	The name of the data source. The name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the data source.
Environment	<p>The type of the environment where the data source will be used.</p> <ul style="list-style-type: none"> ○ If you want to use the data source only in the production environment, select Prod. ○ If you want to use the data source in both the production and development environments, create the data source by using one of the following methods: <ul style="list-style-type: none"> ■ Select Prod and configure the data source for the production environment. After that, go to the Data Sources page, find the data source, and then click Data Source for Dev Environment. ■ Select Prod and Dev and configure the data source for both the production and development environments.
Endpoint	The endpoint that you can use to connect to the RocketMQ instance.
InstanceID	The ID of the RocketMQ instance.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the RocketMQ instance.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the RocketMQ instance.

6. Click **Test Connection** to test the connectivity of the data source.
7. After the data source passes the connectivity test, click **OK**.

9.7.6.4. Modify a data source

This topic describes how to modify a data source.

Prerequisites

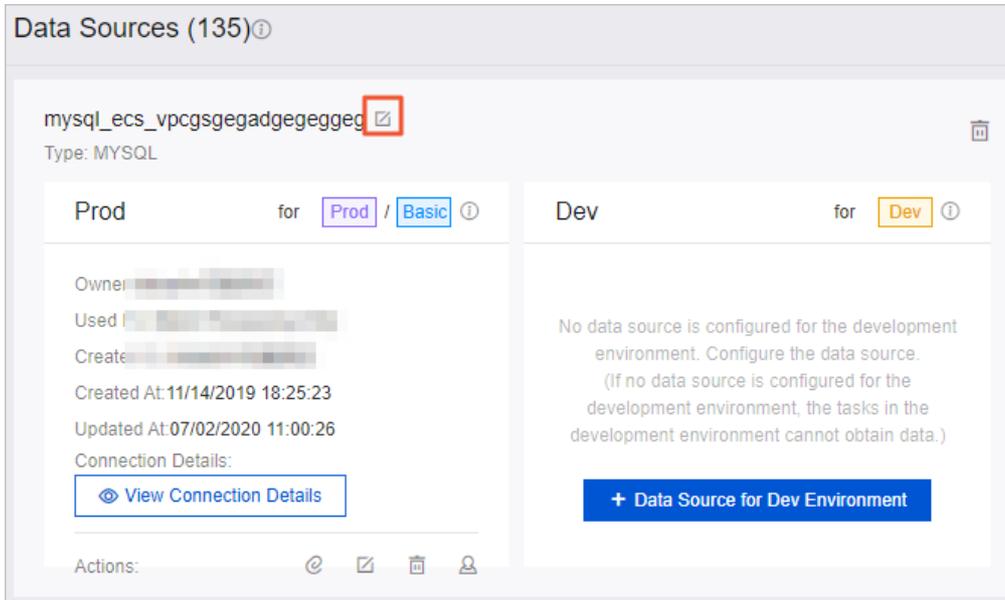
Data sources are created. For more information, see [Create batch processing data sources](#) or [Create stream processing data sources](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.

4. Change the name of a data source.

- i. On the Data Sources page, find the data source that you want to modify and click the  icon next to the name of the data source.



- ii. In the field that appears, enter a new name.

5. Modify the connection information about a data source.

- i. Find the data source that you want to modify and click the  icon next to **Actions** in the Prod or Dev section.
- ii. In the **Edit Data Source for Prod Environment** or **Edit Data Source for Dev Environment** dialog box, change parameter values as needed. For more information about how to set the parameters, see [Create batch processing data sources](#) or [Create stream processing data sources](#).

 **Note** If the data source is bound to a Dataphin project, you cannot change the project name, data source type, or endpoint.

- iii. Click **Test Connection** to test the connectivity of the data source.
- iv. After the data source passes the connectivity test, click **OK**.

9.7.6.5. Test a data source

This topic describes how to test a data source.

Prerequisites

Data sources are created. For more information, see [Create batch processing data sources](#) or [Create stream processing data sources](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - o On the Dataphin homepage, click **Planning** in the top navigation bar.

- On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the Data Sources page, find the data source that you want to test and click the  icon next to **Actions** in the Prod or Dev section.

9.7.6.6. Change the owner of a data source

This topic describes how to change the owner of a data source.

Prerequisites

Data sources are created. For more information, see [Create batch processing data sources](#) or [Create stream processing data sources](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. On the Data Sources page, find the data source that you want to change the owner and click the  icon next to **Actions** in the Prod or Dev section.
5. In the **Change Owner** dialog box, select the member to whom you want to transfer the ownership of the data source.
6. Click OK.

9.7.6.7. Delete a data source

This topic describes how to delete a data source.

Prerequisites

Data sources are created. For more information, see [Create batch processing data sources](#) or [Create stream processing data sources](#).

Context

You cannot delete a data source that is bound to a Dataphin project.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Data Sources** in the left-side navigation pane.
4. Delete a data source.
 - On the Data Sources page, find the data source that you want to delete and click the  icon numbered

2 or 3 in the preceding figure, next to **Actions** in the Prod or Dev section.

- On the Data Sources page, click the  icon numbered 1 in the preceding figure, next to the name of the data source.
5. In the Delete Data Source message, click **OK**.
- If you click the  icon numbered 2 or 3 in the preceding figure, the data source is deleted from the production or development environment.
 - If you click the  icon numbered 1 in the preceding figure, the data source is deleted from both the production and development environments.

9.7.7. Computing engines

9.7.7.1. Overview

A batch processing computing engine provides storage and computing resources for offline computing and a stream processing computing engine provides computing resources for real-time computing.

Usage notes

A stream processing computing engine uses computing resources to process data in queues in real time. A batch processing computing engine can store and then process large amounts of data.

- Before you create a batch processing computing engine, make sure that the computing engine type for batch processing is set on the Computing Engine Configuration page in Management Center.

 **Note** For more information, see [Set the computing engine type](#).

- By default, Dataphin sets Flink as the computing engine type for stream processing.
- Dataphin supports the MaxCompute computing engine for batch processing and the Flink computing engine for stream processing.
- Before you develop tasks in a Dataphin project, you must bind at least one computing engine to the project. By default, a batch processing computing engine is used to store and then process large amounts of data. You can also use a stream processing computing engine to process data in real time.
- Do not create computing engines based on the same connection information. In addition, we recommend that you do not bind a computing engine to multiple Dataphin projects. Otherwise, a conflict may occur when data is written to the physical database.

Permissions

- Only the super administrator, project administrators, and business unit administrators can create computing engines.
- The super administrator can delete and modify all computing engines, test the connectivity of all computing engines, and change the owners of all computing engines.
- Members other than the super administrator can only delete and modify their own computing engines, test the connectivity of their own computing engines, and change the owner of their own computing engines.

9.7.7.2. Create a batch processing computing engine

This topic describes how to create a batch processing computing engine.

Prerequisites

- The computing engine type for batch processing is set. For more information, see [Set the computing engine type](#).
- The endpoint that you can use to connect to MaxCompute is obtained.
- The name of the MaxCompute project to which the computing engine is connected is obtained.
- The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the MaxCompute project is obtained.
- The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the MaxCompute project is obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, move the pointer over **Create Computing Engine** and select **Batch Processing Computing Engine** in the upper-right corner.
5. In the **Batch Processing Computing Engine** dialog box, set the parameters as required.

Parameter	Description
Computing Engine Type	The type of the computing engine. Default value: MaxCompute. You cannot modify this parameter.
Computing Engine Name	The name of the computing engine.
Description	The description of the computing engine.
Endpoint	The endpoint that you can use to connect to MaxCompute. By default, the endpoint is set to the endpoint that is specified on the Computing Engine Configuration page in Management Center. You cannot modify this parameter.
Project Name	The name of the MaxCompute project to which the computing engine is connected.
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the MaxCompute project. Contact the super administrator to obtain the AccessKey ID.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the MaxCompute project. Contact the super administrator to obtain the AccessKey secret.

 **Note**

- You must specify a MaxCompute project that belongs to the same Apsara Stack tenant account as the current account.
- To ensure a successful connection to the MaxCompute project, we recommend that you specify the AccessKey ID and AccessKey secret of the MaxCompute project administrator.
- To ensure normal metadata acquisition, we recommend that you do not change the AccessKey ID and AccessKey secret that are used to connect to the MaxCompute project.
- After a computing engine is created, you cannot change the computing engine type in Dataphin.

6. Click **Test Connection** to test the connectivity of the computing engine.
7. After the computing engine passes the connectivity test, click **Submit**.

9.7.7.3. Create a stream processing computing engine

This topic describes how to create a stream processing computing engine.

Prerequisites

- The name of the Realtime Compute project to which the computing engine is connected is obtained.
- The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the Realtime Compute project is obtained.
- The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the Realtime Compute project is obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, move the pointer over **Create Computing Engine** and select **Stream Processing Computing Engine** in the upper-right corner.
5. In the **Stream Processing Computing Engine** dialog box, set the parameters as required.

Parameter	Description
Computing Engine Type	The type of the computing engine. Default value: FLINK . You cannot modify this parameter.
Computing Engine Name	The name of the computing engine.
Description	The description of the computing engine.
Project Name	The name of the Realtime Compute project to which the computing engine is connected.

Parameter	Description
Access ID	The AccessKey ID that is used to authenticate the user identity when Dataphin connects to the Realtime Compute project.
Access Key	The AccessKey secret that is used to authenticate the user identity when Dataphin connects to the Realtime Compute project.

Note

- You must specify a Realtime Compute project that belongs to the same Apsara Stack tenant account as the current account.
- To ensure a successful connection to the Realtime Compute project, we recommend that you specify the AccessKey ID and AccessKey secret of the Realtime Compute project administrator.
- To ensure normal metadata acquisition, we recommend that you do not change the AccessKey ID and AccessKey secret that are used to connect to the Realtime Compute project.
- After a computing engine is created, you cannot change the computing engine type in Dataphin.

6. Click **Test Connection** to test the connectivity of the computing engine.
7. After the computing engine passes the connectivity test, click **Submit**.

9.7.7.4. Test a computing engine

This topic describes how to test a computing engine.

Prerequisites

Computing engines are created. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, find the computing engine that you want to test and click the  icon in the **Actions** column.

9.7.7.5. Modify a computing engine

This topic describes how to modify a computing engine.

Prerequisites

Computing engines are created. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, find the computing engine that you want to modify and click the  icon in the **Actions** column.
5. In the **Change Data Source (Computing Engine)** dialog box, change the parameter values as needed. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).
6. Click **Test Connection** to test the connectivity of the computing engine.
7. After the computing engine passes the connectivity test, click **Submit**.

9.7.7.6. Change the owner of a computing engine

This topic describes how to change the owner of a computing engine.

Prerequisites

Computing engines are created. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.
 - On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, find the computing engine that you want to change the owner and click the  icon in the **Actions** column.
5. In the **Change Owner** dialog box, select the member to whom you want to transfer the ownership of the computing engine.
6. Click **OK**.

9.7.7.7. Delete a computing engine

This topic describes how to delete a computing engine.

Prerequisites

Computing engines are created. For more information, see [Create a batch processing computing engine](#) or [Create a stream processing computing engine](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Planning** page by using one of the following methods:
 - On the Dataphin homepage, click **Planning** in the top navigation bar.

- On the Dataphin homepage, click **Intelligent Data Warehouse Planning** in the middle section.
3. On the **Planning** page, click **Computing Engines** in the left-side navigation pane.
4. On the Computing Engines page, find the computing engine that you want to delete and click the  icon in the **Actions** column.
5. In the message that appears, click OK.

9.8. Data ingestion

9.8.1. Overview

Dataphin allows you to use the data integration or data synchronization service to import business data from data sources to Dataphin.

If you purchased a Dataphin instance after April 2020, you can use the data integration service to synchronize data, which is simple, efficient, secure, and reliable.

The data integration service allows you to create multiple sync tasks at a time for database migration and can automatically create a destination table for data synchronization to MaxCompute. This improves the data synchronization efficiency.

9.8.2. Data integration

9.8.2.1. Overview

Data integration is a simple, efficient data synchronization service in Dataphin. It provides powerful data preprocessing capabilities and can synchronize data among heterogeneous data sources at a high speed with high stability.

Background information

When big data analytics applies in more and more industries, numerous data integration requirements emerge. For example, people want a platform that can create multiple sync tasks at a time with simple operations, support heterogeneous data sources, preprocess data from data sources, and optimize sync tasks based on fault tolerance, throttling, and concurrency configurations.

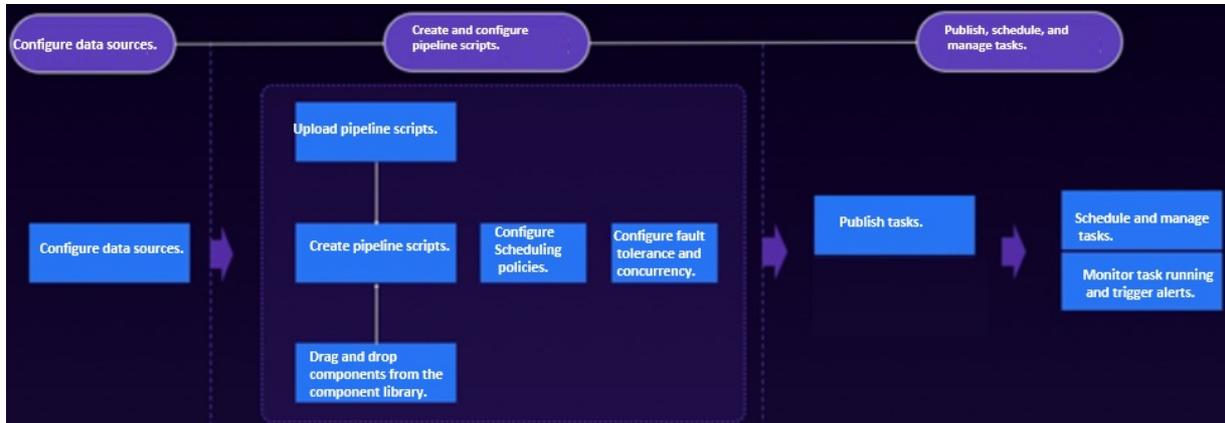
To meet the data integration requirements, Dataphin provides the data integration service with enhanced data integration capabilities. The data integration service is simple, efficient, secure, and reliable and provides the following features:

- Allows you to create multiple sync tasks at a time for database migration and automatically creates a destination table for data synchronization to MaxCompute. This improves the data integration efficiency. For more information, see [Create and configure an offline database migration task](#) and [Manage Maxcompute output components](#).
- Provides process and conversion components to preprocess data from data sources, for example, to cleanse, convert, de-identify, compute, merge, distribute, and filter data. For more information, see [Create and configure an offline migration pipeline](#).
- Synchronizes data among heterogeneous data sources at a high speed with high stability. For more information, see [Create and configure an offline migration pipeline](#) and [Create and configure an offline database migration task](#).
- Supports projects in Dev-Prod mode and Basic mode.
- Synchronizes logical tables that are created in Dataphin to the destination database.

Features

The data integration service allows you to create pipelines by dragging, configuring, and assembling components. You can also configure pipelines, for example, configure the scheduling policy and pipeline information.

The following figure shows the data integration process in Dataphin.



1. Configure data sources. For more information, see [Data sources](#).
2. Create and configure pipelines for single or multiple sync tasks. For more information, see [Create and configure an offline migration pipeline](#) and [Create and configure an offline database migration task](#).
3. Publish tasks. For more information, see [Publishing management](#).
4. Schedule and manage tasks. For more information, see [Overview](#).

9.8.2.2. Manage folders

This topic describes how to create, move, delete, or rename a folder that is used to store pipelines.

Create a folder

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the **Script** section, click the  icon next to **Script**.
6. In the **Create Folder** dialog box, set the **Name** and **Select Directory** parameters.
7. Click **OK**. You can also perform the following steps to create a subfolder under an existing folder:
 - i. Move the pointer over a folder and click the  icon.
 - ii. In the field that appears, enter a folder name.
 - iii. Press the Enter key.

Rename a folder

1. In the **Script** section of the **Integrated** page, move the pointer over the  icon next to the folder that you want to rename and select **Rename**.
2. In the field that appears, enter a new folder name.

3. Press the Enter key.

Move a folder

1. In the Script section of the **Integrated** page, move the pointer over the  icon next to the folder that you want to move and select **Move**.
2. In the **Move Folder** dialog box, set the **Select Directory** parameter to the destination directory.
3. Click **OK**.

Delete a folder

In the Script section of the **Integrated** page, move the pointer over the  icon next to the folder that you want to delete and select **Delete**.

 **Note** You can delete only the folders that do not contain subfolders or items.

9.8.2.3. Upload a JSON file for creating an offline migration pipeline

You can upload a JSON file to Dataphin to create an offline migration pipeline. This topic describes how to upload a JSON file that is used to create an offline migration pipeline.

Prerequisites

A JSON file is prepared.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the **Script** section, click the  icon next to **Script**.
6. In the **Upload Script for Creating Pipeline** dialog box, set the parameters as required.

Parameter	Description
Upload File	The JSON file to upload.
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ○ Recurring Node: a task that is run on a specified schedule. ○ One-Time Node: a task that has no dependencies and is triggered after you click Run.

Parameter	Description
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

7. Click **OK**.
8. (Optional) Configure the scheduling policy.
 - If you set the **Schedule Type** parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Step 3: Configure the scheduling policy](#).
 - If you set the **Schedule Type** parameter to **One-Time Node**, you do not need to configure the scheduling policy.
9. Save, submit, and then publish the offline migration pipeline.
 - i. Click the  icon in the upper-right corner to save the pipeline.
 - ii. Click the  icon in the upper-right corner to submit the pipeline.
 - iii. (Optional) Publish the pipeline.
 - If the current project is in Dev mode, publish the pipeline to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the pipeline after you submit it.

9.8.2.4. Manage offline migration pipelines

9.8.2.4.1. Create and configure an offline migration pipeline

An offline migration pipeline is used to migrate data. This topic describes how to create and configure an offline migration pipeline.

Step 1: Create an offline migration pipeline

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. Move the pointer over **Develop** in the top navigation bar and select **Integrated**.
5. On the **Integrated** page, open the **Create Pipeline Development Script** dialog box by using one of the following methods:
 - In the **Script** section, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - In the left-side navigation pane, move the pointer over the  icon next to the project name and select **Offline Single Pipeline**.
6. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Create Pipeline Development Script
✕

* Pipeline Name

* Schedule Type Recurring Node One-Time Node

Description 0/400

Select Directory offline pipeline

Cancel
OK

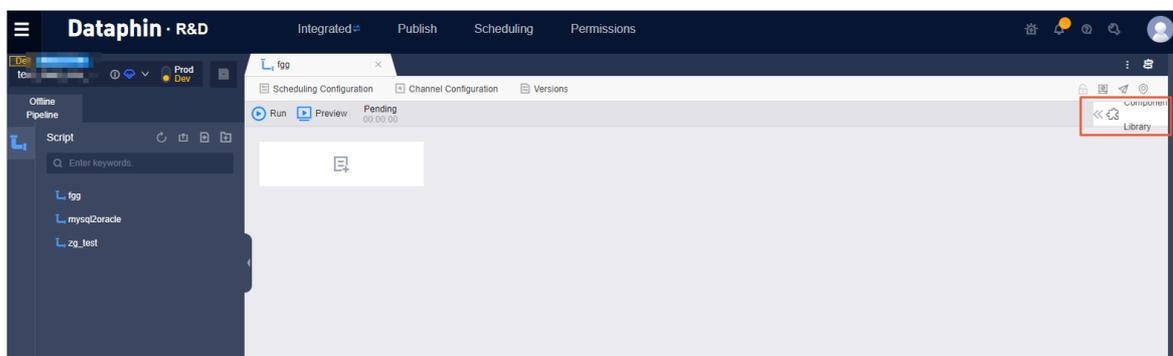
Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ◦ Recurring Node: a task that is run on a specified schedule. ◦ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

7. Click OK.

Step 2: Configure components for the offline migration pipeline

A complete offline migration pipeline is composed of one or more input components, zero or more conversion components, zero or more process components, and one or more output components.

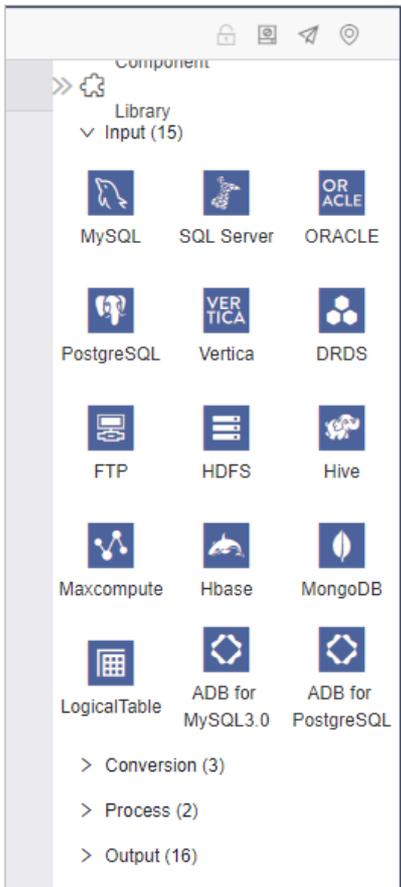
1. Go to the configuration tab of the offline migration pipeline. Click **Component Library** in the upper-right corner. A pane appears and displays the components that Dataphin supports in the **Input**, **Conversion**, **Process**, and **Output** sections.



Component type	Component
----------------	-----------

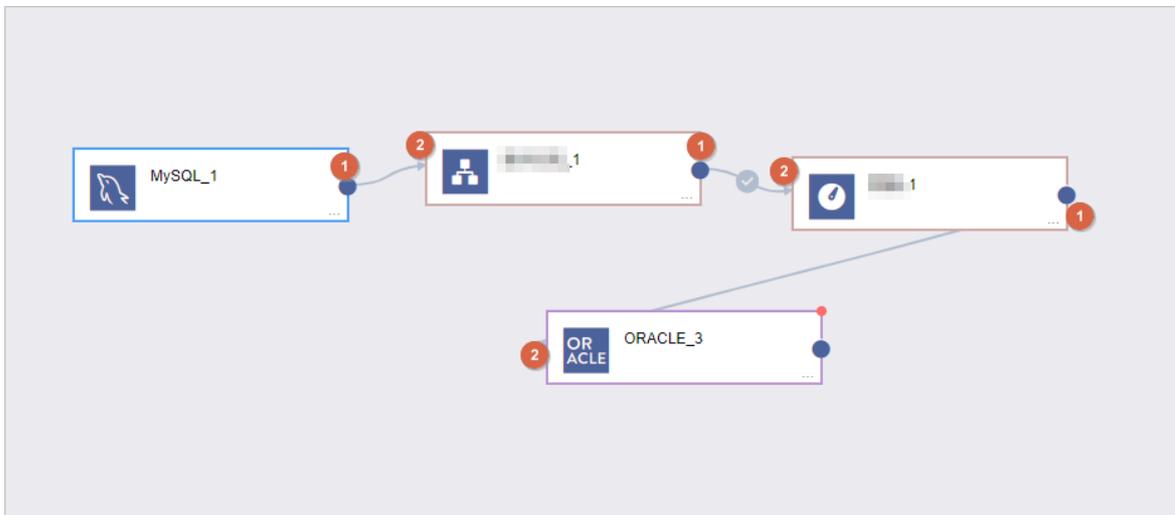
Component type	Component
Input	MySQL, SQL Server, ORACLE, PostgreSQL, Vertica, DRDS, FTP, HDFS, Hive, Maxcompute, Hbase, MongoDB, LogicalTable, ADB for MySQL3.0, and ADB for PostgreSQL
Conversion	Field Selection, Field Computing, and Filtering
Process	Access Limit and Conditional Distribution
Output	MySQL, SQL Server, ORACLE, PostgreSQL, Vertica, DRDS, FTP, HDFS, Hive, Maxcompute, Hbase, MongoDB, ElasticSearch, ADB for MySQL2.0, ADB for MySQL3.0, and ADB for PostgreSQL

2. Select and configure input components based on your business scenario.
 - i. Click the  icon before **Input**. Drag a component to the pipeline canvas on the left.
 - ii. Right-click the component and select **Configure Attributes**. In the dialog box that appears, configure the component as required. For more information, see [Input components](#). You can also copy or delete the component and select a mode for transmitting data to downstream nodes.



3. (Optional) Select and configure conversion components based on your business scenario.
 - i. Click the  icon before **Conversion**. Drag a component to the pipeline canvas on the left.
 - ii. Right-click the component and select **Configure Attributes**. In the dialog box that appears, configure the component as required. For more information, see [Conversion components](#). You can also copy or delete the component and select a mode for transmitting data to downstream nodes.

4. (Optional) Select and configure process components based on your business scenario.
 - i. Click the  icon before **Process**. Drag a component to the pipeline canvas on the left.
 - ii. Right-click the component and select **Configure Attributes**. In the dialog box that appears, configure the component as required. For more information, see [Process components](#). You can also copy or delete the component.
5. Select and configure output components based on your business scenario.
 - i. Click the  icon before **Output**. Drag a component to the pipeline canvas on the left.
 - ii. Right-click the component and select **Configure Attributes**. In the dialog box that appears, configure the component as required. For more information, see [Output components](#). You can also copy or delete the component.
6. Click and hold Position 1 in an upstream component and drag it to Position 2 in a downstream component to draw a directed line and establish a relationship between the components.



The following table describes the relationships that you can establish between components.

Component type	Description
----------------	-------------

Component type	Description
<p>Input</p>	<p>Note the following items when you use an input component:</p> <ul style="list-style-type: none"> ◦ An input component does not support upstream nodes. ◦ The downstream nodes of an input component can be conversion components, output components, and process components. ◦ If you connect an input component to multiple components such as output or conversion components in the downstream, you must select a data transmission mode for the input component. <div data-bbox="572 551 1345 891" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> Set Data Sending Mode × </div> <p style="font-size: 0.9em; margin-top: 5px;">Multiple downstream steps are connected. Select a data sending mode.</p> <p style="text-align: center; margin-top: 5px;"> <input checked="" type="radio"/> Copy <input type="radio"/> Distribute </p> <p style="font-size: 0.8em; margin-top: 5px;">Note: You can also change the data sending mode on the upstream step.</p> <div style="display: flex; justify-content: flex-end; margin-top: 10px;"> Cancel OK </div> </div> <ul style="list-style-type: none"> ▪ Copy: copies all the data of the upstream node to each downstream node. ▪ Distribute: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data shares to the downstream nodes in turn. The sum of the data of all the downstream nodes is equal to that of the upstream node.
<p>Output</p>	<p>An output component does not support downstream nodes.</p>

Component type	Description
Process	<p>Note the following items when you use a process component:</p> <ul style="list-style-type: none"> ○ A process component can be used as any node except the first node and the last node in an offline migration pipeline. ○ If you connect a process component to multiple components such as conversion, output, or process components in the downstream, you must select a data transmission mode for the process component. ○ If you select a Conditional Distribution component as a process component, you must specify the conditions for data distribution when you connect it to downstream nodes. <div data-bbox="571 607 1337 1016" style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p style="text-align: right; margin: 0;">✕</p> <p style="margin: 0;">Set Data Sending Mode</p> <p style="margin: 5px 0 0 0;">If the Conditional Distribution component is used, select the source of data to be sent for the downstream step.</p> <p style="margin: 5px 0 0 20px;"> <input checked="" type="radio"/> Conditioned Result Is true <input type="radio"/> Conditioned Result Is false </p> <p style="margin: 5px 0 0 0; font-size: small;">Note: Only one downstream step is allowed to receive data with the conditioned results of true or false. Only the last downstream step that you selected to connect to the upstream step takes effect.</p> <div style="text-align: right; margin-top: 10px;"> Cancel OK </div> </div> <ul style="list-style-type: none"> ■ If you select Conditioned Result Is true, the Conditional Distribution component transmits data to downstream nodes only when the specified conditions are met. ■ If you select Conditioned Result Is false, the Conditional Distribution component transmits data to downstream nodes only when the specified conditions are not met.
Conversion	<p>If you connect a conversion component to multiple components such as conversion, output, or process components in the downstream, you must select a data transmission mode for the conversion component.</p>

You can perform the preceding steps to configure components for an offline migration pipeline on the pipeline canvas. A pipeline canvas allows you to configure multiple pipelines at the same time. You can also right-click a blank area on the pipeline canvas and perform the operations that are described in the following table.

Operation	Description
Copy	Copy the existing components that you select on the pipeline canvas.
Paste	Paste the copied components on the pipeline canvas.
Delete	Delete the components that you select from the pipeline canvas.
Circle	Select one or more components.
Select All	Select all components on the pipeline canvas.

7. After you configure components for the offline migration pipeline on the pipeline canvas, click the  icon

in the upper-right corner to save the pipeline. Then, the pipeline enters the **Draft** state.

8. Click the  icon before **Pending** in the upper-left corner. If variables such as bizdate are configured in the components, set the parameters as required and click **OK** to run the pipeline.

Step 3: Configure the scheduling policy

1. On the pipeline configuration tab, click **Scheduling Configuration** in the top navigation bar. In the Scheduling Configuration pane, set the parameters as required.
 - i. Set the parameters in the **Basic Information** section. Dataphin automatically generates the node name, node ID, node type, and owner. You cannot modify these parameters.

Parameter	Description
Description	The description of the scheduling policy.
Priority	The priority based on which the task corresponding to the offline migration pipeline is scheduled. Valid values: <ul style="list-style-type: none"> ▪ Lowest Priority ▪ Low Priority ▪ Medium Priority ▪ High Priority ▪ Highest Priority
Parameters	The specified values for the parameters that are used in the code of the task. Dataphin allows you to specify the parameters that are used in the code of a task. The specified values are used when the task is run. You can click Parameters and Descriptions to know how to configure the parameters.

- ii. Set the parameters in the **Scheduling Configuration** section.

Parameter	Description
Schedule Mode	The scheduling mode of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Normal: runs the task based on the specified recurrence. By default, this option is selected. ▪ Dry-run: runs the task based on the specified recurrence. However, the scheduling system does not run the task but returns a success response. ▪ Pause Scheduling: runs the task based on the specified recurrence. However, the scheduling system does not run the task but returns a failure response. You can select this check box if you want to suspend a task and run it later.

Parameter	Description
Recurrence	<p>The recurrence of the task. Valid values:</p> <ul style="list-style-type: none"> Day: automatically runs the task once per day. When you create a recurring task, the task is set to be run at 00:00 every day by default. You can also click the  icon to specify a point in time for the task to be run as needed. Week: automatically runs the task at a specified point in time on specified days of each week. You can also click the  icon to specify a point in time for the task to be run as needed. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not run the instance or consume resources but returns a success response. <p>Assume that you set Recurrence to Week and specify that the task is run every Monday and Tuesday. The scheduling system generates and runs instances every Monday and Tuesday. Every Wednesday, Thursday, Friday, Saturday, and Sunday, the scheduling system generates instances and returns success responses without running the instances.</p> Month: automatically runs the task at a specified point in time on specified days of each month. You can also click the  icon to specify a point in time for the task to be run as needed. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not run the instance or consume resources but returns a success response. <p>Assume that you set Recurrence to Month and specify that the task is run on the seventh day of each month. The scheduling system generates and runs an instance on the seventh day of each month. On the other days, the scheduling system generates instances and returns success responses without running the instances.</p> Hour: automatically runs the task at a specified interval during a specified time period or at specified points in time every day. The scheduling system automatically generates instances for the task and runs the instances at the specified interval or points in time. <p>Assume that you set Recurrence to Hour, select Time Period, and set the Start, End, and Interval parameters to 00:00, 23:59, and 1, respectively. The scheduling system automatically generates instances for the task and runs an instance every hour.</p> Minute: automatically runs the task at a specified interval during a specified time period every day. You can also click the  icon to specify a point in time for the task to be run as needed. <p>Assume that you set Recurrence to Minute and set the Start, End, and Interval parameters to 00:00, 23:59, and 05, respectively. The scheduling system automatically generates instances for the task and runs an instance every 5 minutes.</p>

Parameter	Description
Depend on Previous Instance	<p>Specifies whether the current task is run after the previous instance of another task or of the current task is run. If you select Depend on Previous Instance, you must further select Current Task or Select Task.</p> <ul style="list-style-type: none"> ▪ If you select Current Task, the current task is run after the previous instance of the current task is run. ▪ If you select Select Task, enter a keyword in the search box that appears to search for and select one or more tasks to depend on.

iii. Set the parameters in the **Dependency** section.

Parameter	Description
Upstream Dependency	<p>The upstream nodes on which the current node depends. To specify an upstream node, perform the following steps:</p> <ol style="list-style-type: none"> Click Create Upstream Dependency. In the Create Upstream Dependency dialog box, search for the desired node based on the output name and select the node. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 5px; margin: 5px 0;"> <p> Note Each node output name is globally unique in Dataphin.</p> </div> <ol style="list-style-type: none"> Click OK. <p>To remove a node from the upstream node list, click the  icon in the Actions column.</p>
Current Node	<p>The output name for the current node. You can set multiple output names for a node to be used for the dependency configuration of other nodes. To set an output name, perform the following steps:</p> <ol style="list-style-type: none"> Click Add. In the Add Output Task Nodes for Current Task Node dialog box, enter an output name. Observe uniform rules when you set each output name for the current node. Generally, set an output name in the format of Project name. Table name . This helps other users find this node when they configure the upstream dependency for their nodes. Click OK. <p>You can also perform the following operations on an existing output name:</p> <ul style="list-style-type: none"> ▪ To delete an output name, click the  icon in the Actions column. ▪ To view the downstream nodes after the current node is submitted or published, click the  icon in the Actions column.

2. Click **OK**.

Step 4: Configure the channel control policy

1. On the pipeline configuration tab, click **Channel Configuration** in the top navigation bar. In the Channel Configuration pane, set the parameters as required and click **OK**.

Parameter	Description
-----------	-------------

Parameter	Description
Error Tolerance Configuration	<p>The maximum number of errors allowed in the offline migration pipeline. After you specify a fault tolerance threshold, one of the following situations may occur when the corresponding task is being run:</p> <ul style="list-style-type: none"> ○ The task fails if the number of cumulative errors in all components of the offline migration pipeline reaches the specified fault tolerance threshold. ○ The task continues if the number of cumulative errors in all components of the offline migration pipeline falls below the specified fault tolerance threshold.
Global Concurrency Configuration	The maximum number of components that can concurrently transmit data in the entire offline migration pipeline.
JVM Configuration	The Java virtual machine (JVM) parameters.

2. Click the  icon in the upper-right corner to save the offline migration pipeline.
3. Click the  icon in the upper-right corner to submit the offline migration pipeline.

 **Note** When you submit the pipeline, Dataphin checks whether you have the following permissions:

- Read permission on the data source in each input component
- Write permission on the data source in each output component

4. (Optional) Publish the offline migration pipeline.
 - If the current project is in Dev mode, publish the pipeline to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the pipeline after you submit it.

9.8.2.4.2. Edit an offline migration pipeline

An offline migration pipeline is used to migrate data. This topic describes how to edit an offline migration pipeline.

Prerequisites

Offline migration pipelines are created. For more information, see [Create and configure an offline migration pipeline](#).

Context

To edit an offline migration pipeline, you must be the creator of the pipeline or the super administrator.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.

5. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to edit and select **Change**. The configuration tab of the pipeline appears.
6. (Optional)Steal the lock of the offline migration pipeline.
 - If the pipeline is locked by yourself, you do not need to steal the lock.
 - If the pipeline is locked by another user, click the  icon in the upper-right corner of the pipeline configuration tab to steal the lock.
7. On the pipeline configuration tab, configure the pipeline components as required. For more information, see [Create and configure an offline migration pipeline](#).
8. Save and submit the offline migration pipeline.
 - i. Click the  icon in the upper-right corner of the pipeline configuration tab to save the pipeline.
 - ii. Click the  icon in the upper-right corner of the pipeline configuration tab to submit the pipeline. When you submit the pipeline, Dataphin checks whether you have the following permissions:
 - Read permission on the data source in each input component
 - Write permission on the data source in each output component
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
9. (Optional)Publish the offline migration pipeline.
 - If the current project is in Dev mode, publish the pipeline to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the pipeline after you submit it.

9.8.2.4.3. Rename an offline migration pipeline

An offline migration pipeline is used to migrate data. This topic describes how to rename an offline migration pipeline.

Prerequisites

Offline migration pipelines are created. For more information, see [Create and configure an offline migration pipeline](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional)On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to rename and select **Rename**.
6. In the field that appears, enter a new name.
7. Press the Enter key.

9.8.2.4.4. Move an offline migration pipeline

An offline migration pipeline is used to migrate data. This topic describes how to move an offline migration pipeline to a specified directory.

Prerequisites

Offline migration pipelines are created. For more information, see [Create and configure an offline migration pipeline](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to move and select **Move**.
6. In the **Move File** dialog box, set the **Select Directory** parameter to the destination directory.
7. Click **OK**.

9.8.2.4.5. View historical version information about an offline migration pipeline and download the script in a version of a pipeline

This topic describes how to view historical version information about an offline migration pipeline and download the script in a version of a pipeline.

Prerequisites

Offline migration pipelines are created. For more information, see [Create and configure an offline migration pipeline](#).

View historical version information about an offline migration pipeline

1. [Log on to the Dataphin console](#).
2. Go to the **Integrated** page.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
3. In the left-side navigation pane, click the pipeline that you want to view.
4. On the pipeline configuration tab, click **Versions** in the top navigation bar. If the pipeline is locked by another user, you must click the **Lock** icon in the upper-right corner of the pipeline configuration tab to

steal the lock first.

5. In the **Node Version** pane, view the historical versions of the pipeline.
6. Click the  icon in the **Actions** column of a version to view the version details.
7. On the **Pipeline Details** page, view the components, scheduling configuration, and fault tolerance configuration in the version.
8. Return to the pipeline configuration tab and click **OK** in the **Node Version** pane.

Download the script in a version of an offline migration pipeline

1. In the left-side navigation pane, click the pipeline that you want to view.
2. On the pipeline configuration tab, click **Versions** in the top navigation bar. If the pipeline is locked by another user, you must click the Lock icon in the upper-right corner of the pipeline configuration tab to steal the lock first.
3. In the **Node Version** pane, find a version and click the  icon in the **Actions** column.

9.8.2.4.6. Unpublish and delete an offline migration pipeline

This topic describes how to unpublish, unpublish and delete, and delete offline migration pipelines.

Prerequisites

Offline migration pipelines are created. For more information, see [Create and configure an offline migration pipeline](#).

Context

- Description on the status of an offline migration pipeline:
 - After you create and save a pipeline, it enters the **Draft** state.
 - After you submit a pipeline, it enters the **Submitted** state.
 - After you edit and save a pipeline in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a pipeline in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing offline migration pipelines:
 - You can unpublish only the pipelines in the **Developing** or **Submitted** state.
 - To unpublish a pipeline, you must be the creator of the pipeline or the super administrator.
- Limits on deleting offline migration pipelines:
 - To delete a pipeline, you must be the creator of the pipeline or the super administrator.
 - You can delete only the pipelines in the **Draft** state.
- Limits on unpublishing and deleting offline migration pipelines:
 - You can unpublish and delete only the pipelines in the **Developing** or **Submitted** state.
 - To unpublish and delete a pipeline, you must be the creator of the pipeline or the super administrator.

Unpublish an offline migration pipeline

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.

4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to unpublish and select **Unpublish**.
6. In the **Tip** dialog box, enter your comments.
7. Click **OK**.

Unpublish and delete an offline migration pipeline

1. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to unpublish and delete and select **Unpublish and Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

Delete an offline migration pipeline

1. In the left-side navigation pane, move the pointer over the  icon next to the pipeline that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.8.2.5. Manage offline database migration tasks

9.8.2.5.1. Create and configure an offline database migration task

Dataphin allows you to manage offline database migration tasks. An offline database migration task consists of multiple offline migration pipelines that can migrate multiple tables from one database to another at a time. This topic describes how to create and configure an offline database migration task.

Context

An offline database migration task allows you to migrate multiple tables from one database to another at a time. This improves the configuration efficiency and reduces costs.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. On the **Integrated** page, open the **Create Database Migration** tab by using one of the following methods:
 - In the **Script** section, move the pointer over the  icon next to **Script** and select **Offline Database Migration**.

- o In the left-side navigation pane, move the pointer over the  icon next to the project name and select **Offline Database Migration**.

6. On the **Create Database Migration** tab, set the parameters as required.

- i. Set the parameters in the **Basic Information** section.

Parameter	Description
Script Name	The name of the offline database migration task. The name can be up to 64 characters in length and can contain letters, digits, and underscores (_).
Description	The description of the task.

- ii. Set the parameters in the **Data Source Configuration** section.

Section	Parameter	Description
Synchronization Source	Data Source Type	The type of the data source from which data is read. Valid values: <ul style="list-style-type: none"> ▪ MYSQL ▪ ORACLE ▪ SQL_SERVER
	Data Source	The name of the data source from which data is read. You can also click Create Data Source in the Data Source drop-down list to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source .
Synchronization Target	Data Source Type	The type of the data source to which data is written. You can set this parameter only to MAX_COMPUTE .
	Data Source	The name of the data source to which data is written. You can also click Create Data Source in the Data Source drop-down list to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source .

iii. Set the parameters in the Table Synchronization section.

GUI element	Description
Source Table	The source tables to be migrated.
Target MaxCompute Table	The destination tables to be generated in the specified MaxCompute project. After you select a source table in the Source Table column, a same table name appears in the Target MaxCompute Table column. MaxCompute automatically creates tables based on the table names in the Target MaxCompute Table column.
Configure Conversion	<p>The rules for converting table names and field names. By default, the source tables and destination tables use the same names. This button allows you to modify the configuration of the destination tables. To modify the configuration of the destination tables, perform the following steps:</p> <ol style="list-style-type: none"> a. Click Configure Conversion. b. In the Edit Conversion Rule dialog box, perform one or more of the following operations as required: <ul style="list-style-type: none"> ▪ In the Table Name Conversion section, click Create Rule and set conversion rules for the names of the source and destination tables. ▪ In the Field Name Conversion section, click Create Rule and set conversion rules for the field names in the source and destination tables. ▪ In the Table Name Prefix section, enter a prefix for the names of the destination tables. ▪ In the Data Filtering section, enter a filter condition for input fields, for example, <code>gmt_modified>=\${bizdate}</code>. c. Click Save and Execute. The configured conversion rules apply to the destination tables that are listed in the Target MaxCompute Table column.

iv. Set the parameters in the Synchronization Configuration section.

Parameter	Description
Synchronization Type	The synchronization mode of the task. Valid values: Everyday , Full and Everyday, Incremental .
Synchronization Concurrency	The synchronization concurrency mode of the task. Valid values: Everyday , Full and Batch Upload .

7. Click **Generate Pipeline**. View the pipeline generation result.

8. (Optional) Publish the task.

- If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
- If the current project is in Basic mode, you do not need to publish the task after you submit it.

9.8.2.5.2. Manage offline migration pipelines

After you create an offline database migration task, multiple offline migration pipelines are accordingly generated. This topic describes how to manage an offline migration pipeline that is generated in an offline database migration task.

Prerequisites

Offline database migration tasks are created. For more information, see [Create and configure an offline database migration task](#).

Context

- To edit an offline migration pipeline, you must be the creator of the pipeline or the super administrator.
- Description on the status of an offline migration pipeline:
 - After you create and save a pipeline, it enters the **Draft** state.
 - After you submit a pipeline, it enters the **Submitted** state.
 - After you edit and save a pipeline in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a pipeline in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing offline migration pipelines:
 - You can unpublish only the pipelines in the **Developing** or **Submitted** state.
 - To unpublish a pipeline, you must be the creator of the pipeline or the super administrator.
- Limits on deleting offline migration pipelines:
 - To delete a pipeline, you must be the creator of the pipeline or the super administrator.
 - You can delete only the pipelines in the **Draft** state.
- Limits on unpublishing and deleting offline migration pipelines:
 - You can unpublish and delete only the pipelines in the **Developing** or **Submitted** state.
 - To unpublish and delete a pipeline, you must be the creator of the pipeline or the super administrator.

Edit an offline migration pipeline

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task. In the left-side navigation pane of the **Integrated** page:
 - The  icon indicates a folder where offline migration pipelines reside.
 - The  icon indicates an offline database migration task.
 - The  icon indicates an offline migration pipeline.
6. Move the pointer over the  icon next to the pipeline that you want to edit and select **Change**.
7. (Optional) Steal the lock of the offline migration pipeline.
 - If the pipeline is locked by yourself, you do not need to steal the lock.
 - If the pipeline is locked by another user, click the  icon in the upper-right corner of the pipeline configuration tab to steal the lock.

8. On the pipeline configuration tab, configure the pipeline components as required. For more information, see [Edit an offline migration pipeline](#).
9. Save, submit, and then publish the offline migration pipeline.
 - i. After you edit the pipeline components, click the  icon in the upper-right corner of the pipeline configuration tab to save the pipeline.
 - ii. Click the  icon in the upper-right corner of the pipeline configuration tab to submit the pipeline. When you submit the pipeline, Dataphin checks whether you have the following permissions:
 - Read permission on the data source in each input component
 - Write permission on the data source in each output component
 - iii. Optional. If the current project is in Dev mode, publish the offline migration pipeline to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - After you publish the pipeline to the project in Prod mode, the pipeline can be used for scheduling.
 - If you submit the pipeline in a project in Basic mode, the pipeline can immediately be used for scheduling.

Rename an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. In the left-side navigation pane, move the pointer over  next to the target pipeline and select **Rename**.
3. In the field that appears, enter a new name.
4. Press Enter.

Move an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. In the left-side navigation pane, move the pointer over  next to the target pipeline and select **Move**.
3. In the **Move Folder** dialog box, set the Select Directory parameter to the destination directory.
4. Click **OK**.

View historical version information about an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. In the left-side navigation pane, click the target pipeline.
3. On the pipeline configuration tab, click **Versions** in the top navigation bar. If the pipeline is locked by another user, you must click the Lock icon in the upper-right corner of the pipeline configuration tab to steal the lock first.
4. In the **Node Version** pane, view the historical versions of the pipeline.
5. Click  in the **Actions** column of a version to view the version details.
6. On the **Pipeline Details** page, view the components, scheduling configuration, and error tolerance configuration in the version.
7. Return to the pipeline configuration tab and click **OK** in the **Node Version** pane.

Download the script in a version of an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. Click the pipeline that you want to view.
3. On the pipeline configuration tab, click **Versions** in the top navigation bar. If the pipeline is locked by another user, you must click the Lock icon in the upper-right corner of the pipeline configuration tab to steal the lock first.
4. In the **Node Version** pane, find the target version and click  in the **Actions** column.

Unpublish an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. In the left-side navigation pane, move the pointer over  next to the target pipeline and select **Unpublish**.
3. In the **Tip** dialog box, enter your comments.
4. Click **OK**.

Unpublish and delete an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. Move the pointer over the  icon next to the pipeline that you want to unpublish and delete and select **Unpublish and Delete**.
3. In the **Tip** dialog box, enter your comments.
4. Click **OK**.

Delete an offline migration pipeline

1. On the **Integrated** page, find the offline database migration task that you want to manage and click the  icon before the  icon to show the list of pipelines that are generated in the task.
2. Move the pointer over the  icon next to the pipeline that you want to delete and select **Delete**.
3. In the **Tip** dialog box, enter your comments.
4. Click **OK**.

9.8.2.5.3. Rename an offline database migration task

Dataphin allows you to manage offline database migration tasks. An offline database migration task consists of multiple offline migration pipelines that can migrate multiple tables from one database to another at a time. This topic describes how to rename an offline database migration task.

Prerequisites

Offline database migration tasks are created. For more information, see [Create and configure an offline database migration task](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the left-side navigation pane, move the pointer over the icon next to the offline database migration task that you want to rename.
6. Select **Rename**.
7. In the field that appears, enter a new name.
8. Press the Enter key.

9.8.2.5.4. Delete an offline database migration task

Dataphin allows you to manage offline database migration tasks. An offline database migration task consists of multiple offline migration pipelines that can migrate multiple tables from one database to another at a time. This topic describes how to delete an offline database migration task.

Prerequisites

Offline database migration tasks are created. For more information, see [Create and configure an offline database migration task](#).

Context

If you delete an offline database migration task, all the offline migration pipelines that are generated in the task will also be deleted. Proceed with caution.

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
5. In the left-side navigation pane, move the pointer over the icon next to the offline database migration task that you want to delete.
6. Select **Delete**.

9.8.2.6. Input components

9.8.2.6.1. Manage MySQL input components

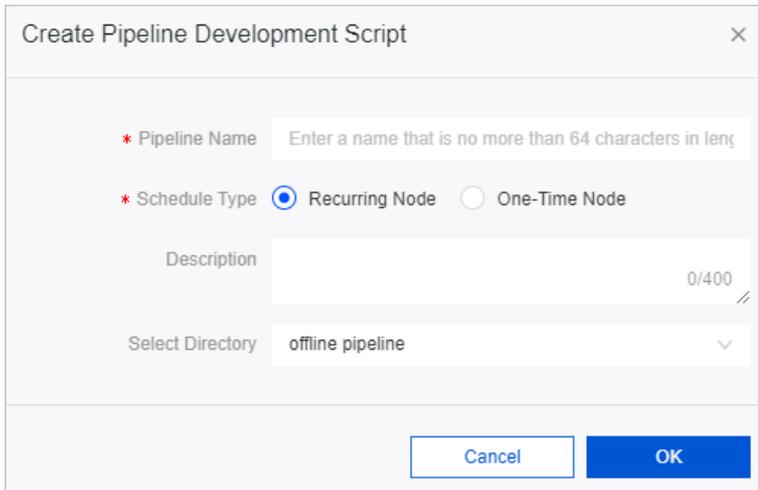
This topic describes how to configure, copy, and delete a MySQL input component, and how to select a data sending mode for a MySQL input component.

Context

A MySQL input component is used to import data from a MySQL database. Then, you can write the data to a database, such as another MySQL database, for data consumption.

Configure a MySQL input component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none">▪ Recurring Node: a task that is run on a specified schedule.▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
 4. Click the  icon before **Input**.

5. Drag a MySQL component to the pipeline canvas on the left.
6. Right-click the MySQL component and select **Configure Attributes**.
7. In the **MySQL Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been connected to Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the MySQL type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Source Table	The quantity of source tables. Valid values: Single Table and Multiple Tables .
Table	<p>The source tables to be migrated.</p> <p>If you set the Source Table parameter to Multiple Tables, perform the following steps to add tables:</p> <ol style="list-style-type: none"> i. Enter an expression in the Table field. Dataphin supports an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>. ii. Click the  icon. iii. In the Confirm Match Details dialog box, select the matching tables. iv. Click OK. <p> Note The source tables that you select must have the same schema.</p>
Split Key	The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key.
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none">Click the  icon in the Actions column to delete a field.Click Manage Fields. In the Manage Fields dialog box, perform the following operations:<ul style="list-style-type: none">Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy a MySQL input component

- Right-click a MySQL input component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a MySQL input component

- Right-click a MySQL input component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for a MySQL input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to downstream nodes.

- Right-click a MySQL input component and select **Data Sending Mode**.
- Select a data sending mode. Two data sending modes are provided:
 - Copy**: copies all the data of the upstream node to each downstream node.
 - Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data shares to the downstream nodes in turn. The sum of the data of all the downstream nodes is equal to that of the upstream node.

9.8.2.6.2. Manage Maxcompute input components

This topic describes how to configure, copy, and delete a Maxcompute input component, and how to select a data sending mode for a Maxcompute input component.

Context

A Maxcompute input component is used to import data from a MaxCompute project. Then, you can write the data to a database, such as a MySQL database, for data consumption.

Configure a Maxcompute input component

- [Log on to the Dataphin console](#).
- Perform the following steps to create an offline migration pipeline:
 - On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a **Maxcompute** component to the pipeline canvas on the left.
6. Right-click the **Maxcompute** component and select **Configure Attributes**.
7. In the **Maxcompute Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been connected to Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the MaxCompute type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	<p>The source table to be migrated. If the table you selected is a partitioned table, enter the partition information, for example, <code>ds=\${bizdate}</code>.</p>
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to delete a field. ◦ Click Manage Fields. In the Manage Fields dialog box, perform the following operations: <ul style="list-style-type: none"> ▪ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ▪ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy a Maxcompute input component

1. Right-click a Maxcompute input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Maxcompute input component

1. Right-click a Maxcompute input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a Maxcompute input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to downstream nodes.

1. Right-click a Maxcompute input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are provided:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data shares to the downstream nodes in turn. The sum of the data of all the downstream nodes is equal to that of the upstream node.

9.8.2.6.3. Manage Vertica input components

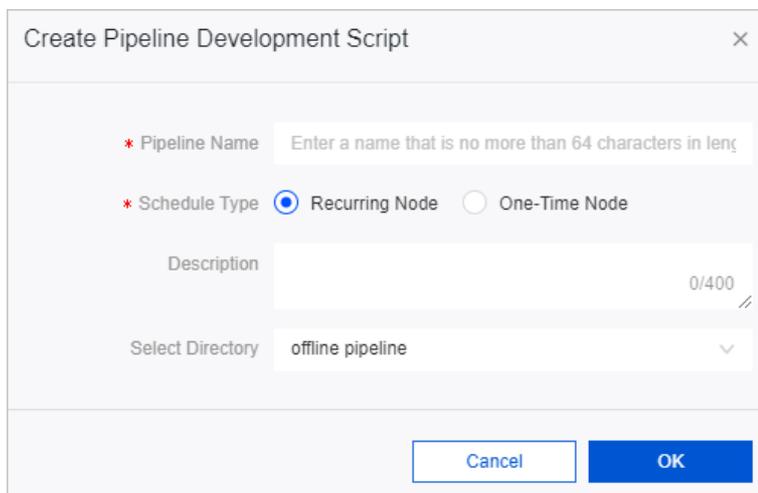
This topic describes how to configure, copy, and delete a Vertica input component, and how to select a data sending mode for a Vertica input component.

Context

A Vertica input component is used to import data from a Vertica database to Dataphin for integration and reprocessing.

Configure a Vertica input component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a Vertica component to the pipeline canvas on the left.
6. Right-click the Vertica component and select **Configure Attributes**.
7. In the **Vertica Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been connected to Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the Vertica type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a Vertica data source.</p>
Table	The source table to be migrated.
Split Key	The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key.
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to delete a field. ◦ Click Manage Fields. In the Manage Fields dialog box, perform the following operations: <ul style="list-style-type: none"> ▪ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ▪ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy a Vertica input component

1. Right-click a Vertica input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Vertica input component

1. Right-click a Vertica input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a Vertica input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to downstream nodes.

1. Right-click a Vertica input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are provided:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data shares to the downstream nodes in turn. The sum of the data of all the downstream nodes is equal to that of the upstream node.

9.8.2.6.4. Manage DRDS input components

This topic describes how to configure, copy, and delete a DRDS input component, and how to select a data sending mode for a DRDS input component.

Context

A DRDS input component is used to import data from a PolarDB-X database to Dataphin for integration and reprocessing.

Configure a DRDS input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a DRDS component to the pipeline canvas on the left.
6. Right-click the DRDS component and select **Configure Attributes**.
7. In the **DRDS Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the PolarDB-X type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The source table.
Split Key	The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key.
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to delete a field. ◦ Click Manage Fields. In the Manage Fields dialog box, perform the following operations: <ul style="list-style-type: none"> ▪ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ▪ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy a DRDS input component

1. Right-click a DRDS input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a DRDS input component

1. Right-click a DRDS input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a DRDS input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click a DRDS input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - o **Copy**: copies all the data of the upstream node to each downstream node.
 - o **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.5. Manage PostgreSQL input components

This topic describes how to configure, copy, and delete a PostgreSQL input component, and how to select a data sending mode for a PostgreSQL input component.

Context

A PostgreSQL input component is used to import data from a PostgreSQL database to Dataphin for integration and reprocessing.

Configure a PostgreSQL input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a PostgreSQL component to the pipeline canvas on the left.
6. Right-click the PostgreSQL component and select **Configure Attributes**.
7. In the **PostgreSQL Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the PostgreSQL type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a PostgreSQL data source.</p>
Source Table	<p>The quantity of source tables. Valid values: Single Table and Multiple Tables.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note If you select Multiple Tables, you must specify multiple source tables that have the same schema.</p> </div>
Table	<p>The one or more source tables.</p> <ul style="list-style-type: none"> ◦ If you set Source Table to Single Table, click the  icon and select a source table from the Table drop-down list. ◦ If you set Source Table to Multiple Tables, perform the following steps to specify the source tables: <ul style="list-style-type: none"> a. Enter an expression in the Table field. Dataphin allows you to enter an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>. b. Click the  icon. c. In the Confirm Match Details dialog box, select the tables that match the expression you entered. d. Click OK.
Split Key	<p>The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key. If you specify multiple source tables, you need to specify only one shard key because the source tables have the same schema.</p>
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ○ Click the  icon in the Actions column to delete a field. ○ Click Manage Fields. In the Manage Fields dialog box, view the unselected and selected input fields and perform the following operations: <ul style="list-style-type: none"> ■ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ■ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy a PostgreSQL input component

1. Right-click a PostgreSQL input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a PostgreSQL input component

1. Right-click a PostgreSQL input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a PostgreSQL input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click a PostgreSQL input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.6. Manage SQL Server input components

This topic describes how to configure, copy, and delete an SQL Server input component, and how to select a data sending mode for an SQL Server input component.

Context

An SQL Server input component is used to import data from an SQL Server database to Dataphin for integration and reprocessing.

Configure an SQL Server input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an **SQL Server** component to the pipeline canvas on the left.
6. Right-click the **SQL Server** component and select **Configure Attributes**.
7. In the **SQL Server Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> The data source is of the SQL Server type. The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Source Table	<p>The quantity of source tables. Valid values: Single Table and Multiple Tables.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note If you select Multiple Tables, you must specify multiple source tables that have the same schema.</p> </div>
Table	<p>The one or more source tables.</p> <ul style="list-style-type: none"> If you set Source Table to Single Table, click the  icon and select a source table from the Table drop-down list. If you set Source Table to Multiple Tables, perform the following steps to specify the source tables: <ul style="list-style-type: none"> Enter an expression in the Table field. <p>Dataphin allows you to enter an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>.</p> Click the  icon. In the Confirm Match Details dialog box, select the tables that match the expression you entered. Click OK.
Split Key	<p>The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key. If you specify multiple source tables, you need to specify only one shard key because the source tables have the same schema.</p>
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> Specify a fixed portion of data. Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none">Click the  icon in the Actions column to delete a field.Click Manage Fields. In the Manage Fields dialog box, perform the following operations:<ul style="list-style-type: none">Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy an SQL Server input component

- Right-click an SQL Server input component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an SQL Server input component

- Right-click an SQL Server input component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for an SQL Server input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

- Right-click an SQL Server input component and select **Data Sending Mode**.
- Select a data sending mode. Two data sending modes are supported:
 - Copy**: copies all the data of the upstream node to each downstream node.
 - Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.7. Manage ORACLE input components

This topic describes how to configure, copy, and delete an ORACLE input component, and how to select a data sending mode for an ORACLE input component.

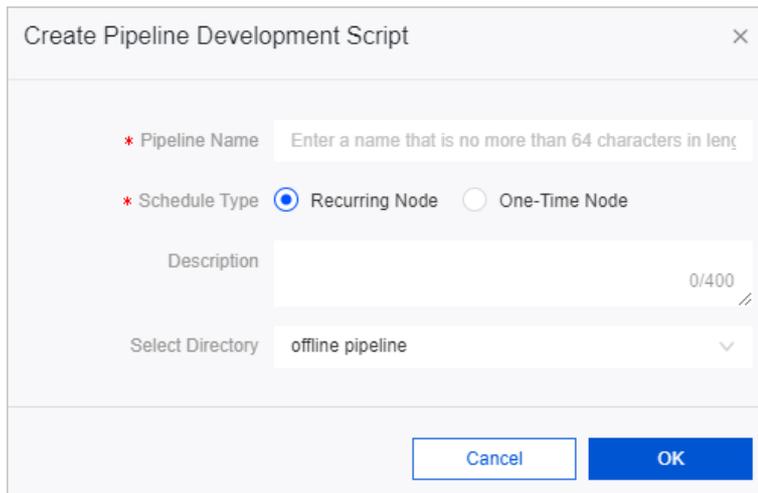
Context

An ORACLE input component is used to import data from an Oracle database to Dataphin for integration and reprocessing.

Configure an ORACLE input component

- [Log on to the Dataphin console](#).
- Perform the following steps to create an offline migration pipeline:

- i. On the Dataphin homepage, click **R&D** in the top navigation bar.
- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an Oracle component to the pipeline canvas on the left.
6. Right-click the Oracle component and select **Configure Attributes**.
7. In the **Oracle Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the Oracle type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an Oracle data source.</p>
Source Table	<p>The quantity of source tables. Valid values: Single Table and Multiple Tables.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note If you select Multiple Tables, you must specify multiple source tables that have the same schema.</p> </div>
Table	<p>The one or more source tables.</p> <ul style="list-style-type: none"> ◦ If you set Source Table to Single Table, click the  icon and select a source table from the Table drop-down list. ◦ If you set Source Table to Multiple Tables, perform the following steps to specify the source tables: <ul style="list-style-type: none"> a. Enter an expression in the Table field. Dataphin allows you to enter an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>. b. Click the  icon. c. In the Confirm Match Details dialog box, select the tables that match the expression you entered. d. Click OK.
Split Key	<p>The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key. If you specify multiple source tables, you need to specify only one shard key because the source tables have the same schema.</p>
Encoding	<p>The encoding format of the source table. Valid values: UTF-8, GBK, and ISO-8859-1.</p>
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ○ Click the  icon in the Actions column to delete a field. ○ Click Manage Fields. In the Manage Fields dialog box, perform the following operations: <ul style="list-style-type: none"> ▪ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ▪ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy an ORACLE input component

1. Right-click an ORACLE input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ORACLE input component

1. Right-click an ORACLE input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for an ORACLE input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click an ORACLE input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.8. Manage FTP input components

This topic describes how to configure, copy, and delete an FTP input component, and how to select a data sending mode for an FTP input component.

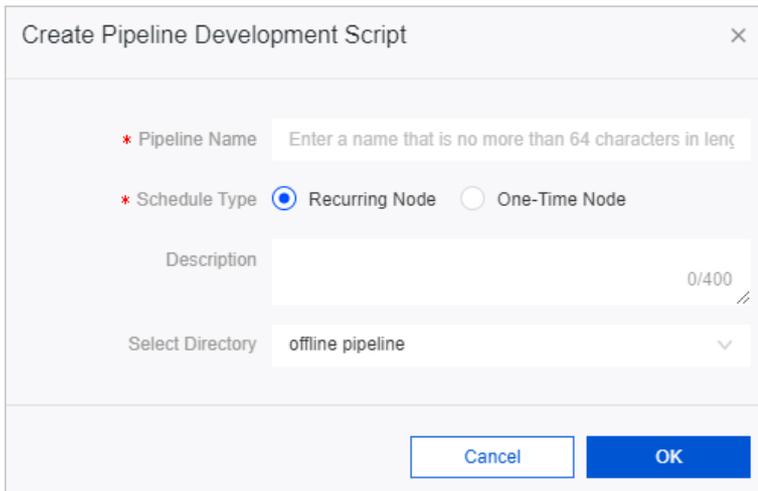
Context

An FTP input component is used to import data from an FTP server to Dataphin for integration and reprocessing.

Configure an FTP input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an FTP component to the pipeline canvas on the left.
6. Right-click the FTP component and select **Configure Attributes**.
7. In the **FTP Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the FTP or SFTP type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an FTP data source.</p>
File Path	The path of the source file.
File Type	The type of the source file. Valid values: Text and CSV .
File Encoding	The encoding format of the source file. Valid values: UTF-8 and GBK .
Field Delimiter	The delimiter that is used to separate fields in the source file. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Compression Format	<p>The compression format of the source file. Valid values:</p> <ul style="list-style-type: none"> ◦ zip ◦ gzip ◦ bzip2
First Line Type	<p>The content type of the first line of the source file.</p> <ul style="list-style-type: none"> ◦ If you select Data Content, you must configure output fields by using one of the following methods: <ul style="list-style-type: none"> ▪ Click Create Output Field. Set the Source Serial Number, Field, and Type parameters as required. ▪ Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. ◦ If you select Field Name, the output fields are automatically parsed based on the fields in the first line of the source file.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <div style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <pre>[[{ "index": 0, "name": "user_id", "type": "String" }, { "index": 1, "name": "user_name", "type": "String" }]]</pre> </div> <p>In the example, index, name, and type specify the sequence number, name, and data type of the output field, respectively.</p> Click Create Output Field. Set the Source Serial Number, Field, and Type parameters as required. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy an FTP input component

- Right-click an FTP input component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an FTP input component

- Right-click an FTP input component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for an FTP input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

- Right-click an FTP input component and select **Data Sending Mode**.
- Select a data sending mode. Two data sending modes are supported:
 - Copy**: copies all the data of the upstream node to each downstream node.
 - Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.9. Manage HDFS input components

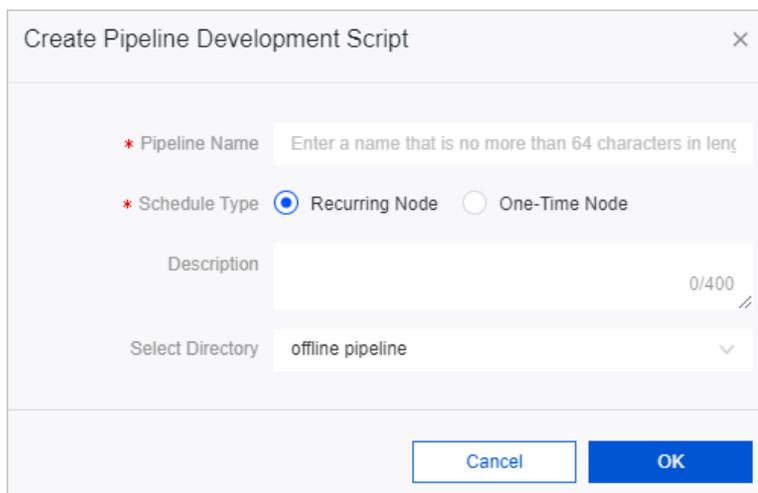
This topic describes how to configure, copy, and delete an HDFS input component, and how to select a data sending mode for an HDFS input component.

Context

An HDFS input component is used to import data from an HDFS to Dataphin for integration and reprocessing.

Configure an HDFS input component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.

4. Click the  icon before **Input**.
5. Drag an HDFS component to the pipeline canvas on the left.
6. Right-click the HDFS component and select **Configure Attributes**.
7. In the **HDFS Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the HDFS type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an HDFS data source.</p>
File Path	The path of the source file.
File Type	The type of the source file. Valid values: Text , CSV , ORC , RC , and Sequence .
File Encoding	The encoding format of the source file. Valid values: UTF-8 and GBK .
Field Delimiter	The delimiter that is used to separate fields in the source file. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Compression Format	<p>The compression format of the source file. Valid values:</p> <ul style="list-style-type: none"> ◦ zip ◦ gzip ◦ bzip2

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <div data-bbox="576 434 1385 772" style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <pre> [[{ "index": 0, "name": "user_id", "type": "String" }, { "index": 1, "name": "user_name", "type": "String" }]] </pre> </div> <p>In the example, index, name, and type specify the sequence number, name, and data type of the output field, respectively.</p> Click Create Output Field. Set the Source Serial Number, Field, and Type parameters as required. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy an HDFS input component

- Right-click an HDFS input component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an HDFS input component

- Right-click an HDFS input component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for an HDFS input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

- Right-click an HDFS input component and select **Data Sending Mode**.
- Select a data sending mode. Two data sending modes are supported:
 - Copy**: copies all the data of the upstream node to each downstream node.
 - Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.10. Manage Hive input components

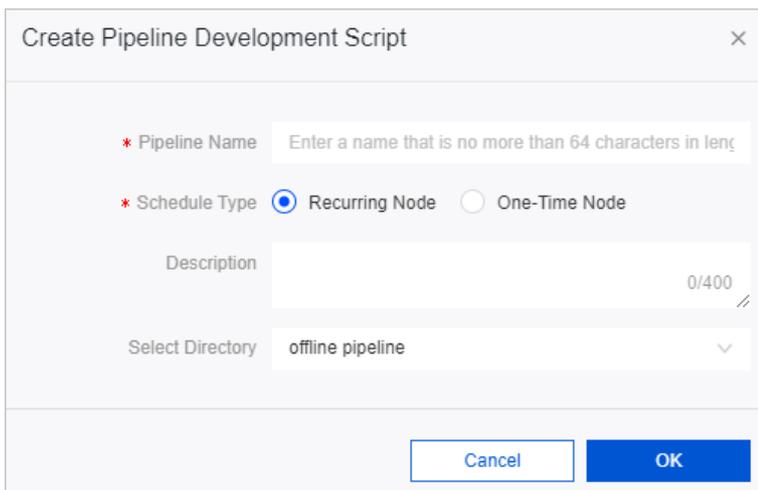
This topic describes how to configure, copy, and delete a Hive input component, and how to select a data sending mode for a Hive input component.

Context

A Hive input component is used to import data from a Hive database to Dataphin for integration and reprocessing.

Configure a Hive input component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none">▪ Recurring Node: a task that is run on a specified schedule.▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a Hive component to the pipeline canvas on the left.
6. Right-click the Hive component and select **Configure Attributes**.
7. In the **Hive Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the Hive type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The source table.
File Encoding	The encoding format of the source table. Valid values: UTF-8 and GBK .
Compression Format	<p>The format that is used to compress the source file. Valid values:</p> <ul style="list-style-type: none"> ◦ zip ◦ gzip ◦ bzip2
Field Delimiter	The delimiter that is used to separate fields in the source table. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Output Fields	<p>The output fields to be generated based on the input configuration. You can click Manage Fields and perform the following operations in the Manage Fields dialog box:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. <p>You can also click the  icon in the Actions column to delete a field.</p>

8. Click **OK**.

Copy a Hive input component

1. Right-click a Hive input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Hive input component

1. Right-click a Hive input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a Hive input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click a Hive input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.11. Manage Hbase input components

This topic describes how to configure, copy, and delete an Hbase input component, and how to select a data sending mode for an Hbase input component.

Context

An Hbase input component is used to import data from an HBase database to Dataphin for integration and reprocessing.

Configure an Hbase input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an Hbase component to the pipeline canvas on the left.
6. Right-click the Hbase component and select **Configure Attributes**.
7. In the **HBase Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the HBase type. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an HBase data source.</p>
Table	The source table.
Mode	<p>The mode of the source table. Valid values: Normal Mode and Multiple Versions.</p> <p>If you select Multiple Versions, you must set the Version parameter to specify the versions of data to read. If you set the Version parameter to -1, all versions of data will be read.</p>
File Encoding	The encoding format of the source table. Valid values: UTF-8 and GBK .
Input Filtering	The filter condition for input fields. For example, you can enter <code>"startRowkey":"20190101000000"</code> .
Output Fields	<p>The output fields to be generated based on the input configuration. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> ◦ Click Create Output Field. Set the Column Family, Field, and Type parameters as required. ◦ Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format and click OK. Example: <pre> [[{ "index": 0, "name": "user_id", "type": "String" }, { "index": 1, "name": "user_name", "type": "String" }]] </pre> <p>In the example, index, name, and type specify the sequence number, name, and data type of the output field, respectively.</p> <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to delete a field. ◦ Click the  icon in the Actions column to edit a field.

8. Click **OK**.

Copy an Hbase input component

1. Right-click an Hbase input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an Hbase input component

1. Right-click an Hbase input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for an Hbase input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click an Hbase input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.12. Manage MongoDB input components

This topic describes how to configure, copy, and delete a MongoDB input component, and how to select a data sending mode for a MongoDB input component.

Context

A MongoDB input component is used to import data from a MongoDB database to Dataphin for integration and reprocessing.

Configure a MongoDB input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a MongoDB component to the pipeline canvas on the left.
6. Right-click the MongoDB component and select **Configure Attributes**.
7. In the **MongoDB Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> The data source is of the MongoDB type. The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The source table.
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> Specify a fixed portion of data. Use parameters to filter data.
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format and click OK. Example: <pre>[[{"name":"cf1:a","type":"String"}, {"name":"cf1:b","type":"String"}]]</pre> Click Create Output Field. Set the Field and Type parameters as required. You can also perform the following operations on existing fields: <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy a MongoDB input component

- Right-click a MongoDB input component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a MongoDB input component

- Right-click a MongoDB input component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for a MongoDB input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

- Right-click a MongoDB input component and select **Data Sending Mode**.

2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.13. Manage LogicalTable input components

This topic describes how to configure, copy, and delete a LogicalTable input component, and how to select a data sending mode for a LogicalTable input component.

Context

A LogicalTable input component is used to import data from a logical table created in Dataphin to a database for integration and reprocessing.

Configure a LogicalTable input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

The screenshot shows a dialog box titled "Create Pipeline Development Script". It has a close button (X) in the top right corner. The form contains the following elements:

- * Pipeline Name:** A text input field with a placeholder "Enter a name that is no more than 64 characters in length".
- * Schedule Type:** Two radio buttons: "Recurring Node" (selected) and "One-Time Node".
- Description:** A text input field with a character count "0/400" and a clear icon.
- Select Directory:** A dropdown menu currently showing "offline pipeline".

At the bottom of the dialog, there are two buttons: "Cancel" and "OK".

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a **LogicalTable** component to the pipeline canvas on the left.
6. Right-click the **LogicalTable** component and select **Configure Attributes**.
7. In the **Logical Table Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Business Units	The business unit where the logical table resides.
Logical Table Type	The type of the logical table. Valid values: <ul style="list-style-type: none"> ◦ Logical Aggregate Table ◦ Logical Dimension Table ◦ Logical Fact Table

Parameter	Description
Logical Tables	The logical table. The system displays the logical tables on which your account has the read-through permission. You can also click the  icon to go to the Develop page and create a logical table.
Logical Table Output Mode	The output mode of the logical table. Valid values: <ul style="list-style-type: none"> ◦ Without Association: The output fields to be generated are the fields of the selected logical table. ◦ With Association: The output fields to be generated are the fields of the selected logical table and associated table.
Output Fields	The output fields to be generated. You can perform the following steps to preview the output fields: <ol style="list-style-type: none"> Click Manage Fields. Select one or more fields in the All Fields section. The output fields to be generated appear in the Output Field Preview section. <p>You can also click the  icon in the Actions column to delete a field.</p>

8. Click OK.

Copy a LogicalTable input component

1. Right-click a LogicalTable input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a LogicalTable input component

1. Right-click a LogicalTable input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a LogicalTable input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click a LogicalTable input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy:** copies all the data of the upstream node to each downstream node.
 - **Distribute:** divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.14. Manage ADB for MySQL3.0 input components

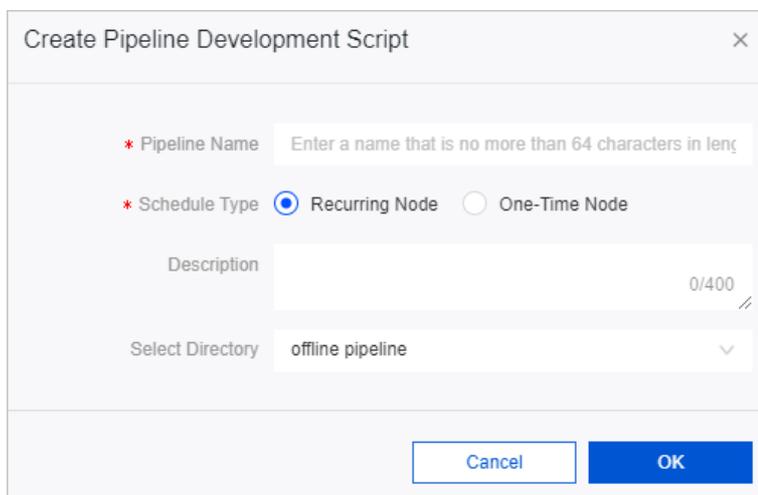
This topic describes how to configure, copy, and delete an ADB for MySQL3.0 input component, and how to select a data sending mode for an ADB for MySQL3.0 input component.

Context

An ADB for MySQL3.0 input component is used to import data from an AnalyticDB for MySQL 3.0 database to Dataphin for integration and reprocessing.

Configure an ADB for MySQL3.0 input component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an ADB for MySQL3.0 component to the pipeline canvas on the left.
6. Right-click the ADB for MySQL3.0 component and select **Configure Attributes**.
7. In the **ADB for MySQL 3.0 Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the input component. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an AnalyticDB for MySQL V3.0 data source.</p>
Source Table	<p>The quantity of source tables. Valid values: Single Table and Multiple Tables.</p> <div style="background-color: #e1f5fe; padding: 5px; border: 1px solid #bbdefb;"> <p> Note If you select Multiple Tables, you must specify multiple source tables that have the same schema.</p> </div>
Table	<p>The one or more source tables. Specify one or more source tables based on the Source Table parameter.</p> <ul style="list-style-type: none"> ◦ If you set Source Table to Single Table, click the  icon and select a source table from the Table drop-down list. ◦ If you set Source Table to Multiple Tables, perform the following steps to specify the source tables: <ul style="list-style-type: none"> a. Enter an expression in the Table field. Dataphin allows you to enter an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>. b. Click the  icon. c. In the Confirm Match Details dialog box, select the tables that match the expression you entered. d. Click OK.
Split Key	The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key. If you specify multiple source tables, you need to specify only one shard key because the source tables have the same schema.
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none"> ○ Click the  icon in the Actions column to delete a field. ○ Click Manage Fields. In the Manage Fields dialog box, perform the following operations: <ul style="list-style-type: none"> ■ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ■ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy an ADB for MySQL3.0 input component

1. Right-click an ADB for MySQL3.0 input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ADB for MySQL3.0 input component

1. Right-click an ADB for MySQL3.0 input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for an ADB for MySQL3.0 input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click an ADB for MySQL3.0 input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.15. Manage ADB for PostgreSQL input components

This topic describes how to configure, copy, and delete an ADB for PostgreSQL input component, and how to select a data sending mode for an ADB for PostgreSQL input component.

Context

An ADB for PostgreSQL input component is used to import data from an AnalyticDB for PostgreSQL database to Dataphin for integration and reprocessing.

Configure an ADB for PostgreSQL input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag an ADB for PostgreSQL component to the pipeline canvas on the left.
6. Right-click the ADB for PostgreSQL component and select **Configure Attributes**.
7. In the **ADB for PostgreSQL Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the input component. ◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an AnalyticDB for PostgreSQL data source.</p>
Source Table	<p>The quantity of source tables. Valid values: Single Table and Multiple Tables.</p> <div style="background-color: #e6f2ff; padding: 5px; border: 1px solid #d9e1f2;"> <p> Note If you select Multiple Tables, you must specify multiple source tables that have the same schema.</p> </div>
Table	<p>The one or more source tables. Specify one or more source tables based on the Source Table parameter.</p> <ul style="list-style-type: none"> ◦ If you set the Source Table parameter to Single Table, click the  icon and select a source table from the Table drop-down list. ◦ If you set Source Table to Multiple Tables, perform the following steps to specify the source tables: <ul style="list-style-type: none"> a. Enter an expression in the Table field. Dataphin allows you to enter an enumeration expression, a regular-like expression, or a combination of both, for example, <code>table_[001-100];table_102</code>. b. Click the  icon. c. In the Confirm Match Details dialog box, select the tables that match the expression you entered. d. Click OK.
Split Key	<p>The shard key of the source table. You can specify a column in the source table as the shard key. We recommend that you use the primary key or an indexed column as the shard key. If you specify multiple source tables, you need to specify only one shard key because the source tables have the same schema.</p>
Input Filtering	<p>The filter condition for input fields. For example, you can enter <code>ds=\${bizdate}</code>. The Input Filtering parameter is applicable to the following scenarios:</p> <ul style="list-style-type: none"> ◦ Specify a fixed portion of data. ◦ Use parameters to filter data.

Parameter	Description
Output Fields	<p>The output fields to be generated based on the input configuration. You can also perform the following operations to manage fields:</p> <ul style="list-style-type: none">○ Click the  icon in the Actions column to delete a field.○ Click Manage Fields. In the Manage Fields dialog box, perform the following operations:<ul style="list-style-type: none">■ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.■ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.

8. Click **OK**.

Copy an ADB for PostgreSQL input component

1. Right-click an ADB for PostgreSQL input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ADB for PostgreSQL input component

1. Right-click an ADB for PostgreSQL input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for an ADB for PostgreSQL input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click an ADB for PostgreSQL input component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.6.16. Manage LogHub input components

This topic describes how to configure, copy, and delete a LogHub input component, and how to select a data sending mode for a LogHub input component.

Context

A LogHub input component is used to import data from a LogHub data source to Dataphin for integration and reprocessing.

Configure a LogHub input component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Input**.
5. Drag a LogHub component to the pipeline canvas on the left.
6. Right-click the LogHub component and select **Configure Attributes**.
7. In the **Loghub Input Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none">◦ The data source is of the LogHub type.◦ The account that you use to configure the component has the read-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Logstore	<p>The Logstore where the source data resides. All data in the Logstore is from the same data source.</p>
Log StartTime	<p>The start time of the period in which the required logs were generated. The value is in the format <code>yyyyMMddHHmmss</code>.</p>
Log EndTime	<p>The end time of the period in which the required logs were generated. The value is in the format <code>yyyyMMddHHmmss</code>.</p>
Batch Number	<p>The number of data records to read at a time. You can enter a number in the Batch Number field. Alternatively, you can click the upwards or downwards arrow in the field to adjust the value. Default value: 256. The system can read up to 1,000 data records at a time.</p>

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format and click OK. Example: <div data-bbox="576 434 1385 772" style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <pre> [[{ "index": 0, "name": "user_id", "type": "String" }, { "index": 1, "name": "user_name", "type": "String" }]] </pre> </div> <p>In the example, index, name, and type specify the sequence number, name, and data type of the output field, respectively.</p> Click Create Output Field. Set the Field, Source Serial Number, and Type parameters as required. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy a LogHub input component

1. Right-click a LogHub input component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a LogHub input component

1. Right-click a LogHub input component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a LogHub input component

If an input component is connected to multiple components in the downstream, you must select a mode in which the data of the input component is sent to the downstream nodes.

1. Right-click a LogHub input component and select **Data Sending Mode**.

2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.7. Conversion components

9.8.2.7.1. Manage Field Selection components

This topic describes how to configure, copy, and delete a Field Selection component, and how to select a data sending mode for a Field Selection component.

Context

A Field Selection component is used to filter the input fields from an upstream component, rename the fields, and change the order of output fields.

Configure a Field Selection component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

- On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
- Click the  icon before **Conversion**.
- Drag a Field Selection component to the pipeline canvas on the left.
- Right-click the Field Selection component and select **Configure Attributes**.
- In the **Field Selection Configuration** dialog box, set the parameter as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Input Fields	The input fields based on the upstream input.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none">◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section.◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none">◦ Click the  icon in the Actions column to edit a field.◦ Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy a Field Selection component

1. Right-click a Field Selection component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Field Selection component

1. Right-click a Field Selection component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a Field Selection component

If a conversion component is connected to multiple components in the downstream, you must select a mode in which the data of the conversion component is sent to the downstream nodes.

1. Right-click a Field Selection component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:
 - **Copy**: copies all the data of the upstream node to each downstream node.
 - **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.7.2. Manage Field Computing components

This topic describes how to configure, copy, and delete a Field Computing component, and how to select a data sending mode for a Field Computing component.

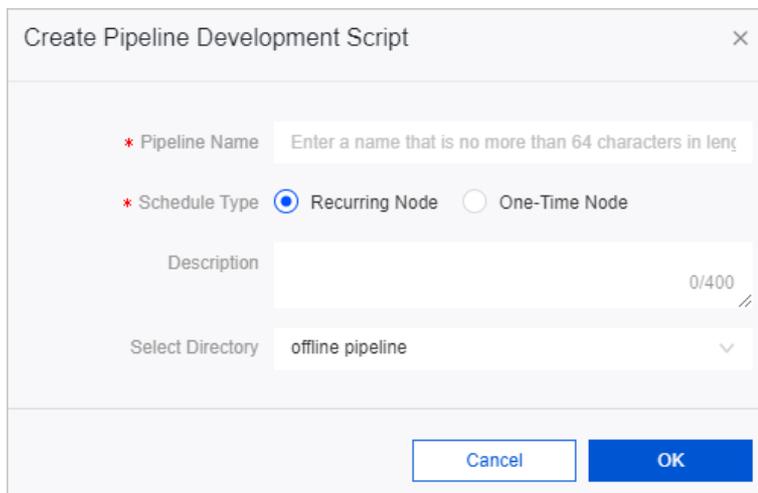
Context

The applicable scenarios of a Field Computing component include but are not limited to:

- Split an input field from an upstream component into multiple fields.
- Generate derived fields based on the input fields from an upstream component.
- Map or de-identify the values of the input fields from an upstream component.

Configure a Field Computing component

1. Log on to the Dataphin console.
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Conversion**.
5. Drag a Field Computing component to the pipeline canvas on the left.
6. Right-click the Field Computing component and select **Configure Attributes**.
7. In the **Field Computing Configuration** dialog box, set the parameter as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. To create an output field, perform the following steps:</p> <ol style="list-style-type: none">Click Create Field.In the Create Field dialog box, set the Name, Expression, and Data Type parameters. When you set the Expression parameter, you can view the functions in the system and their usage in the Function section on the right.Click OK. <p>To delete an output field, click the  icon in the Actions column.</p>

8. Click **OK**.

Copy a Field Computing component

- Right-click a Field Computing component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Field Computing component

- Right-click a Field Computing component and select **Delete**.
- In the message that appears, click **OK**.

Select a data sending mode for a Field Computing component

If a conversion component is connected to multiple components in the downstream, you must select a mode in which the data of the conversion component is sent to the downstream nodes.

- Right-click a Field Computing component and select **Data Sending Mode**.
- Select a data sending mode. Two data sending modes are supported:
 - Copy**: copies all the data of the upstream node to each downstream node.
 - Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.7.3. Manage Filtering components

This topic describes how to configure, copy, and delete a Filtering component, and how to select a data sending mode for a Filtering component.

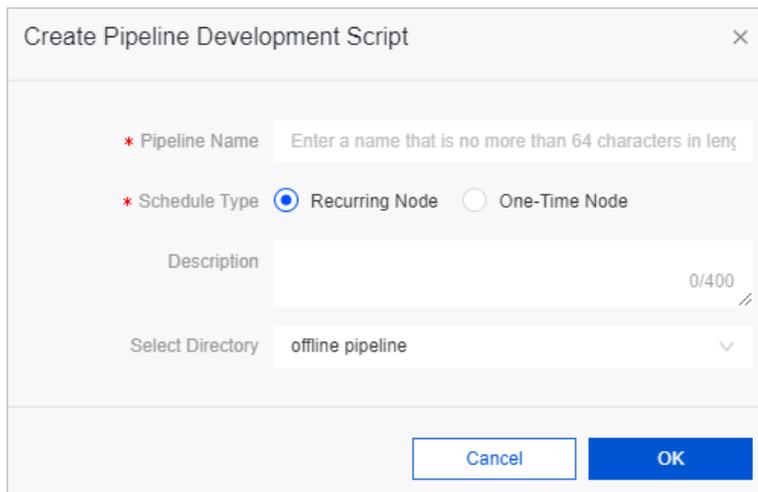
Context

A Filtering component is used to filter the input data from an upstream component and send the data that meets the specified filter conditions to output components.

Configure a Filtering component

- [Log on to the Dataphin console](#).

2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Conversion**.
5. Drag a **Filtering** component to the pipeline canvas on the left.
6. Right-click the **Filtering** component and select **Configure Attributes**.
7. In the **Conversion-Filtering Configuration** dialog box, set the parameter as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Input Fields	The input fields based on the upstream input.
Filter Conditions	<p>The filter conditions for the input data. You can specify the filter conditions by using one of the following methods:</p> <ul style="list-style-type: none"> ○ Perform the following steps: <ul style="list-style-type: none"> a. Click the  icon in the Field column and select a field from the drop-down list. b. Click the  icon in the Operator column and select an operator from the drop-down list. The following operators are supported: =, !=, >, >=, <, <=, LINKE, IS NULL, and IS NOT NULL. c. Click the  icon in the Content column and select Table Field or Custom Recipients from the drop-down list. <ul style="list-style-type: none"> ■ If you select Table Field, another drop-down list appears in the Content column. Select a field from the drop-down list. ■ If you select Custom Recipients, a field appears in the Content column. Enter a recipient in the field. Click the  icon and then the  icon. Select a data type from the drop-down list. d. Click the  icon in the Actions column. The  icon appears in a new line. Click this icon and select AND or OR from the drop-down list. ○ Click Switch to Script Mode and enter filter conditions in the code editor that appears. For example, you can enter <code>user_id is not null and user_name like '%a%'</code>. The following operators are supported: =, !=, >, >=, <, <=, LIKE, IS NULL, and IS NOT NULL.
Output Fields	The output fields to be generated.

8. Click **OK**.

Copy a Filtering component

1. Right-click a Filtering component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Filtering component

1. Right-click a Filtering component and select **Delete**.
2. In the message that appears, click **OK**.

Select a data sending mode for a Filtering component

If a conversion component is connected to multiple components in the downstream, you must select a mode in which the data of the conversion component is sent to the downstream nodes.

1. Right-click a Filtering component and select **Data Sending Mode**.
2. Select a data sending mode. Two data sending modes are supported:

- **Copy**: copies all the data of the upstream node to each downstream node.
- **Distribute**: divides the data of the upstream node into equal shares based on the number of downstream nodes and distributes the data to the downstream nodes in turn. The total data volume of all the downstream nodes is equal to the data volume of the upstream node.

9.8.2.8. Process components

9.8.2.8.1. Manage Access Limit components

This topic describes how to configure, copy, and delete an Access Limit component.

Context

An Access Limit component is used to control read/write operations on downstream databases.

Configure an Access Limit component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

The dialog box titled "Create Pipeline Development Script" has a close button (X) in the top right. It contains the following fields:

- * Pipeline Name:** A text input field with a placeholder "Enter a name that is no more than 64 characters in length".
- * Schedule Type:** Two radio buttons: "Recurring Node" (selected) and "One-Time Node".
- Description:** A text input field with a character count "0/400" and a slash icon.
- Select Directory:** A dropdown menu showing "offline pipeline".

At the bottom, there are "Cancel" and "OK" buttons.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
- 3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
- 4. Click the icon before **Process**.
- 5. Drag an **Access Limit** component to the pipeline canvas on the left.
- 6. Right-click the **Access Limit** component and select **Configure Attributes**.
- 7. In the **Access Limit Configuration** dialog box, set the parameter as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Speed Limit	The upper limit of the data transmission speed. <ul style="list-style-type: none"> ○ If you select Data Volume-based Limit, you can select a maximum data transmission rate. Valid values: 1M/s, 2M/s, 5M/s, and 10M/s. ○ If you select Data Record-based Limit, you must enter the maximum number of data records that can be sent per second, for example, 30 records per second.

8. Click **OK**.

Copy an Access Limit component

1. Right-click an Access Limit component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an Access Limit component

1. Right-click an Access Limit component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.8.2. Manage Conditional Distribution components

This topic describes how to configure, copy, and delete a Conditional Distribution component.

Context

A Conditional Distribution component is used to split the input data from an upstream component and distribute the data to different downstream components.

Configure a Conditional Distribution component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the icon before **Process**.
5. Drag a **Conditional Distribution** component to the pipeline canvas on the left.
6. Right-click the **Conditional Distribution** component and select **Configure Attributes**.
7. In the **Conditional Distribution Configuration** dialog box, set the parameter as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Input Fields	The input fields based on the upstream input.
Merge Logic Configuration	The conditions for distribution. The following operators are supported: =, !=, >, >=, <, <=, LIKE, IN, IS NULL, and IS NOT NULL. For example, you can enter <code>user_id is not null and user_name like '%a%'</code> . Distribution is only based on the results of true and false. If you do not specify the conditions for distribution, the result defaults to true.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section. ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to edit a field. ◦ Click the  icon in the Actions column to delete a field.

8. Click **OK**.

Copy a Conditional Distribution component

1. Right-click a Conditional Distribution component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Conditional Distribution component

1. Right-click a Conditional Distribution component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9. Output components

9.8.2.9.1. Manage MySQL output components

This topic describes how to configure, copy, and delete a MySQL output component.

Context

A MySQL output component is used to write data processed by Dataphin to a MySQL database for data consumption.

Configure a MySQL output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
 4. Click the  icon before **Output**.
 5. Drag a MySQL component to the pipeline canvas on the left.
 6. Right-click the MySQL component and select **Configure Attributes**.
 7. In the **MySQL Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source.</p>
Table	The destination table.
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Parse Solution	Optional. The operations before and after data output. For more information, move the pointer over Component Description in the upper-right corner of the dialog box.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click OK.

Copy a MySQL output component

1. Right-click a MySQL output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a MySQL output component

1. Right-click a MySQL output component and select **Delete**.
2. In the message that appears, click OK.

9.8.2.9.2. Manage DRDS output components

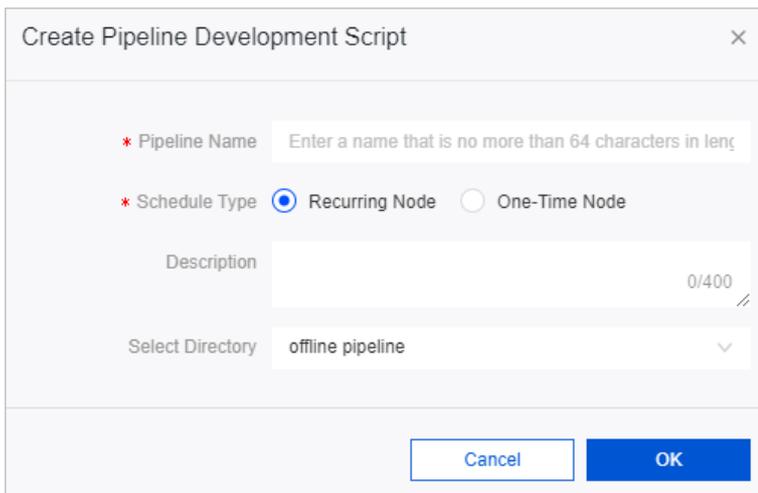
This topic describes how to configure, copy, and delete a DRDS output component.

Context

A DRDS output component is used to write data processed by Dataphin to a PolarDB-X database for data consumption.

Configure a DRDS output component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none">▪ Recurring Node: a task that is run on a specified schedule.▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.

4. Click the  icon before **Output**.
5. Drag a DRDS component to the pipeline canvas on the left.
6. Right-click the DRDS component and select **Configure Attributes**.
7. In the **DRDS Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Parse Solution	Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy a DRDS output component

1. Right-click a DRDS output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a DRDS output component

1. Right-click a DRDS output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.3. Manage Maxcompute output components

This topic describes how to configure, copy, and delete a Maxcompute output component.

Context

A Maxcompute output component is used to write data processed by Dataphin to a MaxCompute project for data consumption.

Configure a Maxcompute output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag a Maxcompute component to the pipeline canvas on the left.
6. Right-click the Maxcompute component and select **Configure Attributes**.
7. In the **Maxcompute Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	<p>The destination table. You can also perform the following steps to create a destination table:</p> <ol style="list-style-type: none"> Click Generate Target Table by One Click. In the dialog box that appears, write an SQL script as required and click Create. <p>By default, the table created is the destination table for the output data.</p>
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Partitioning	<p>The partition information required if the destination table is a partitioned table.</p> <ul style="list-style-type: none"> ◦ If the table you selected is a partitioned table, you must enter partition information, for example, <code>ds=\${bizdate}</code>. ◦ If the table you selected is not a partitioned table, you do not need to set this parameter.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click OK.

Copy a Maxcompute output component

1. Right-click a Maxcompute output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Maxcompute output component

1. Right-click a Maxcompute output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.4. Manage SQL Server output components

This topic describes how to configure, copy, and delete an SQL Server output component.

Context

An SQL Server output component is used to write data processed by Dataphin to an SQL Server database for data consumption.

Configure an SQL Server output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
- 3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
- 4. Click the  icon before **Output**.
- 5. Drag an SQL Server component to the pipeline canvas on the left.
- 6. Right-click the SQL Server component and select **Configure Attributes**.
- 7. In the **SQL Server Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> The data source is of the same type as the output component. The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> Enter Preparation Statement: the SQL statements to execute before data output. Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p> <ul style="list-style-type: none"> Position-based Mapping: The fields in the Input Fields section and those in the Output Fields section are mapped based on their positions. Name-based Mapping: The fields in the Input Fields section and those in the Output Fields section are mapped based on their names.

8. Click OK.

Copy an SQL Server output component

- Right-click an SQL Server output component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an SQL Server output component

1. Right-click an SQL Server output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.5. Manage ORACLE output components

This topic describes how to configure, copy, and delete an ORACLE output component.

Context

An ORACLE output component is used to write data processed by Dataphin to an Oracle database for data consumption.

Configure an ORACLE output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an ORACLE component to the pipeline canvas on the left.
6. Right-click the ORACLE component and select **Configure Attributes**.
7. In the **Oracle Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the properties has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> ◦ Enter Preparation Statement: the SQL statements to execute before data output. ◦ Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy an ORACLE output component

1. Right-click an ORACLE output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ORACLE output component

1. Right-click an ORACLE output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.6. Manage PostgreSQL output components

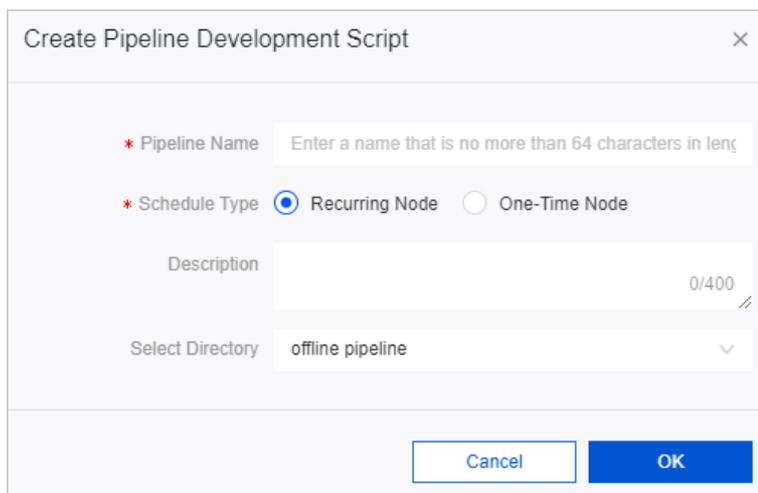
This topic describes how to configure, copy, and delete a PostgreSQL output component.

Context

A PostgreSQL output component is used to write data processed by Dataphin to a PostgreSQL database for data consumption.

Configure a PostgreSQL output component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.

4. Click the  icon before **Output**.
5. Drag a PostgreSQL component to the pipeline canvas on the left.
6. Right-click the PostgreSQL component and select **Configure Attributes**.
7. In the **PostgreSQL Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> ◦ Enter Preparation Statement: the SQL statements to execute before data output. ◦ Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy a PostgreSQL output component

1. Right-click a PostgreSQL output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a PostgreSQL output component

1. Right-click a PostgreSQL output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.7. Manage Vertica output components

This topic describes how to configure, copy, and delete a Vertica output component.

Context

A Vertica output component is used to write data processed by Dataphin to a Vertica database for data consumption.

Configure a Vertica output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

The screenshot shows a dialog box titled "Create Pipeline Development Script" with a close button (X) in the top right corner. The dialog contains the following fields and options:

- * Pipeline Name:** A text input field with a placeholder "Enter a name that is no more than 64 characters in length".
- * Schedule Type:** Two radio button options: "Recurring Node" (which is selected) and "One-Time Node".
- Description:** A text input field with a character count "0/400" and a double-slash icon.
- Select Directory:** A dropdown menu currently showing "offline pipeline".

At the bottom of the dialog, there are two buttons: "Cancel" and "OK".

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag a Vertica component to the pipeline canvas on the left.
6. Right-click the Vertica component and select **Configure Attributes**.
7. In the **Vertica Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> ◦ Enter Preparation Statement: the SQL statements to execute before data output. ◦ Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy a Vertica output component

1. Right-click a Vertica output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Vertica output component

1. Right-click a Vertica output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.8. Manage FTP output components

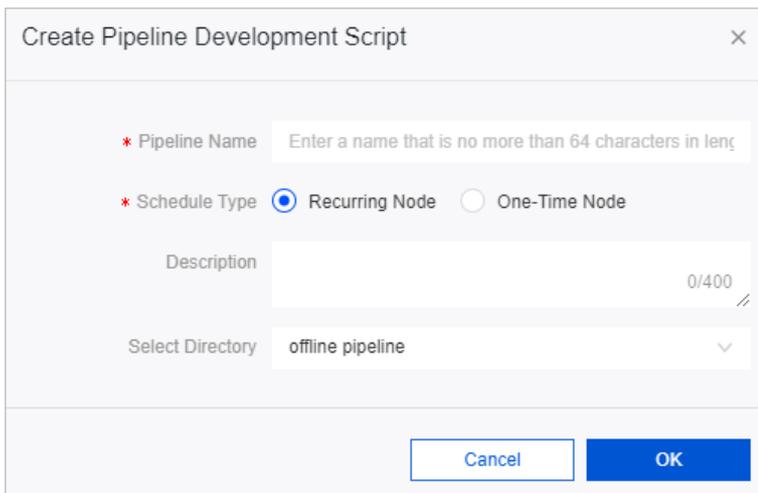
This topic describes how to configure, copy, and delete an FTP output component.

Context

An FTP output component is used to write data processed by Dataphin to an FTP server for data consumption.

Configure an FTP output component

1. [Log on to the Dataphin console.](#)
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none">▪ Recurring Node: a task that is run on a specified schedule.▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.

4. Click the  icon before **Output**.
5. Drag an FTP component to the pipeline canvas on the left.
6. Right-click the FTP component and select **Configure Attributes**.
7. In the **FTP Output Configuration** dialog box, set the parameters as required.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
File Path	The path of the destination file.
File Type	The type of the destination file. Valid values: Text and CSV .
File Encoding	The encoding format of the destination file. Valid values: UTF-8 and GBK .
Field Delimiter	Optional. The delimiter that is used to separate fields in the destination file. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Loading Policy	<p>The policy for writing data to the destination file. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination file. ◦ Append Data: appends the source data to the destination file, without modifying the existing data in the destination file.
Input Fields	The input fields based on the upstream input.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none">Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example:<pre>[[{"name":"cf1:a","type":"String"}, {"name":"cf1:b","type":"String"}]]</pre>Click Create Output Field. Set the Field and Type parameters as required. You can also perform the following operations on existing fields:<ul style="list-style-type: none">Click the  icon in the Actions column to edit a field.Click the  icon in the Actions column to delete a field.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click **OK**.

Copy an FTP output component

1. Right-click an FTP output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an FTP output component

1. Right-click an FTP output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.9. Manage HDFS output components

This topic describes how to configure, copy, and delete an HDFS output component.

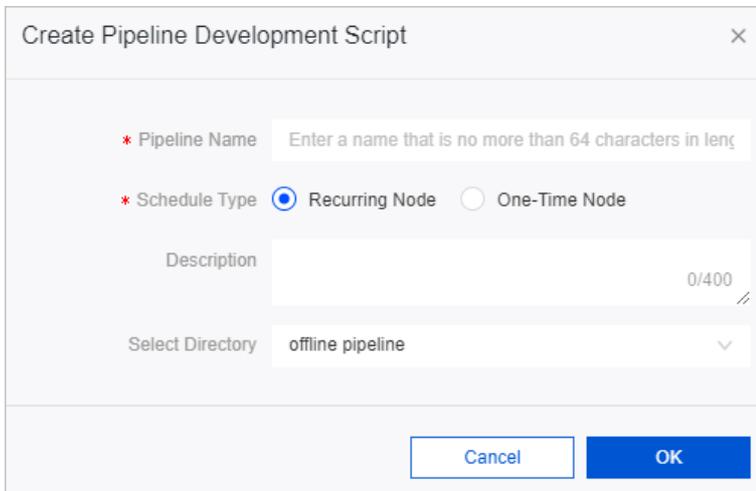
Context

An HDFS output component is used to write data processed by Dataphin to an HDFS for data consumption.

Configure an HDFS output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.

- iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
- v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an HDFS component to the pipeline canvas on the left.
6. Right-click the HDFS component and select **Configure Attributes**.
7. In the **HDFS Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
File Path	The path of the destination file.
File Type	The type of the destination file. Valid values: Text and ORC .
File Encoding	The encoding format of the destination file. Valid values: UTF-8 and GBK .
Field Delimiter	Optional. The delimiter that is used to separate fields in the destination file. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Loading Policy	<p>The policy for writing data to the destination file. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination file. ◦ Append Data: appends the source data to the destination file, without modifying the existing data in the destination file.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> ◦ Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <pre> [[{"name":"cf1:a","type":"String"}, {"name":"cf1:b","type":"String"}]] </pre> ◦ Click Create Output Field. Set the Field and Type parameters as prompted. You can also perform the following operations on existing fields: <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to edit a field. ◦ Click the  icon in the Actions column to delete a field.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click OK.

Copy an HDFS output component

1. Right-click an HDFS output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an HDFS output component

1. Right-click an HDFS output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.10. Manage MongoDB output components

This topic describes how to configure, copy, and delete a MongoDB output component.

Context

A MongoDB output component is used to write data processed by Dataphin to a MongoDB database for data consumption.

Configure a MongoDB output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag a MongoDB component to the pipeline canvas on the left.
6. Right-click the MongoDB component and select **Configure Attributes**.
7. In the **MongoDB Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MongoDB data source.</p>
Table	The destination table.
Update Information	The update information of fields. For example, you can enter <code>{"isUpsert":"true", "upsertkey":"unique_id"}</code> .
Field Delimiter	The delimiter that is used to separate fields in the destination table. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> ◦ Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <pre>[[{"name":"cf1:a","type":"String"}, {"name":"cf1:b","type":"String"}]]</pre> ◦ Click Create Output Field. Set the Field and Type parameters as prompted. You can also perform the following operations on existing fields: <ul style="list-style-type: none"> ◦ Click the  icon in the Actions column to edit a field. ◦ Click the  icon in the Actions column to delete a field.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy a MongoDB output component

1. Right-click a MongoDB output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a MongoDB output component

1. Right-click a MongoDB output component and select **Delete**.

2. In the message that appears, click **OK**.

9.8.2.9.11. Manage Hbase output components

This topic describes how to configure, copy, and delete an Hbase output component.

Context

An Hbase output component is used to write data processed by Dataphin to an HBase database for data consumption.

Configure an Hbase output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an Hbase component to the pipeline canvas on the left.
6. Right-click the Hbase component and select **Configure Attributes**.
7. In the **HBase Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
File Encoding	The encoding format of the destination table. Valid values: UTF-8 and GBK .
Version Column	The timestamp of data written to the destination table in HBase. For example, you can enter {"index":-1,"value":123456789}.
Rowkey	<p>The rowkey rule of the destination table. The rowkey is composed of several output fields that are connected by an underscore (_). Example:</p> <pre>[{"index":0,"type":"string"},{"index":2,"type":"string"},{"index":-1,"string","value":"_"}]</pre> <p>In this example, the rowkey is composed of the first and third output fields that are connected by an underscore (_).</p>
Input Fields	The input fields based on the upstream input.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <pre> [[{ "name": "col_integer", "type": "integer" }, { "name": "col_long", "type": "long" }, { "name": "col_double", "type": "double" }]] </pre> <p>In the example, name and type specify the name and data type of the output field, respectively.</p> Click Create Output Field. Set the Column Family, Field, and Type parameters as prompted. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click **OK**.

Copy an Hbase output component

- Right-click an Hbase output component and select **Copy**.
- Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an Hbase output component

- Right-click an Hbase output component and select **Delete**.
- In the message that appears, click **OK**.

9.8.2.9.12. Manage Hive output components

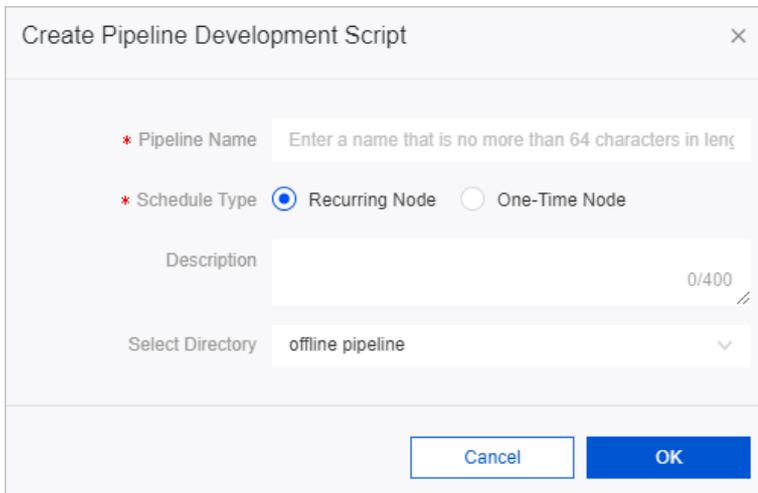
This topic describes how to configure, copy, and delete a Hive output component.

Context

A Hive output component is used to write data processed by Dataphin to a Hive database for data consumption.

Configure a Hive component

1. Log on to the Dataphin console.
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag a Hive component to the pipeline canvas on the left.
6. Right-click the Hive component and select **Configure Attributes**.
7. In the **Hive Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a Hive data source.</p>
Table	The destination table.
File Encoding	The encoding format of the destination table. Valid values: UTF-8 and GBK .
Field Delimiter	Optional. The delimiter that is used to separate fields in the destination table. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Compression Format	<p>The compression format of the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ zip ◦ gzip ◦ bzip2
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click OK.

Copy a Hive component

1. Right-click a Hive component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete a Hive component

1. Right-click a Hive component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.13. Manage Elasticsearch output components

This topic describes how to configure, copy, and delete an Elasticsearch output component.

Context

An Elasticsearch output component is used to write data processed by Dataphin to an Elasticsearch database for data consumption.

Configure an Elasticsearch output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

The screenshot shows a dialog box titled "Create Pipeline Development Script". It has a close button (X) in the top right corner. The form contains the following elements:

- * Pipeline Name:** A text input field with a placeholder "Enter a name that is no more than 64 characters in length".
- * Schedule Type:** Two radio buttons: "Recurring Node" (selected) and "One-Time Node".
- Description:** A text input field with a character count "0/400" and a double-slash icon.
- Select Directory:** A dropdown menu showing "offline pipeline".

At the bottom of the dialog, there are two buttons: "Cancel" and "OK".

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an **ElasticSearch** component to the pipeline canvas on the left.
6. Right-click the **ElasticSearch** component and select **Configure Attributes**.
7. In the **ElasticSearch Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Index	The index of the destination table.
Index Type	The index type of the destination table.
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Field Delimiter	Optional. The delimiter that is used to separate fields in the destination table. If you do not specify this parameter, a comma (,) is used as the field delimiter.
Input Fields	The input fields based on the upstream input.

Parameter	Description
Output Fields	<p>The output fields to be generated. You can create output fields by using one of the following methods:</p> <ul style="list-style-type: none"> Click Batch Create. In the Batch Create in JSON Format dialog box, configure multiple output fields in the JSON format. Example: <pre> [[{ "name": "col_integer", "type": "integer" }, { "name": "col_long", "type": "long" }, { "name": "col_double", "type": "double" }]] </pre> <p>In the example, name and type specify the name and data type of the output field, respectively.</p> Click Create Output Field. Set the Field and Type parameters as required. <p>You can also perform the following operations on existing fields:</p> <ul style="list-style-type: none"> Click the  icon in the Actions column to edit a field. Click the  icon in the Actions column to delete a field.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click **OK**.

Copy an Elasticsearch output component

1. Right-click an Elasticsearch output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an Elasticsearch output component

1. Right-click an Elasticsearch output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.14. Manage ADB for MySQL2.0 output components

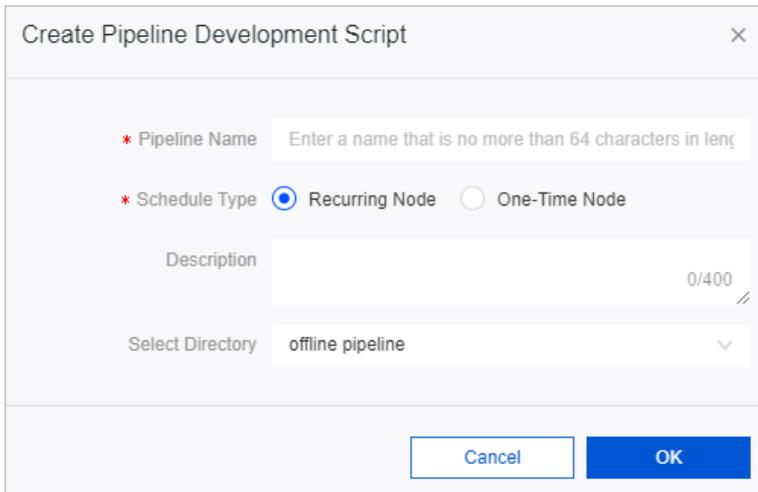
This topic describes how to configure, copy, and delete an ADB for MySQL2.0 output component.

Context

An ADB for MySQL2.0 output component is used to write data processed by Dataphin to an AnalyticDB for MySQL 2.0 database for data consumption.

Configure an ADB for MySQL2.0 output component

1. Log on to the Dataphin console.
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.
 - v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.



Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

- vi. Click **OK**.
3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an ADB for MySQL2.0 component to the pipeline canvas on the left.
6. Right-click the ADB for MySQL2.0 component and select **Configure Attributes**.
7. In the **AnalyticDB for MySQL 2.0 Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create an AnalyticDB for MySQL V3.0 data source.</p>
Table	The destination table.
Mode	<p>The mode for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Insert Mode: applicable when fewer than 10 million data records are to be written. If you select this mode, perform the following operations to set the Parse Solution parameter: <ul style="list-style-type: none"> ▪ Click Enter Preparation Statement and enter the SQL statements to execute before data output. ▪ Click Enter Completion Statement and enter the SQL statements to execute after data output. ◦ Load Mode: applicable when more than 10 million data records are to be written. If you select this mode, you must set the following parameters: <ul style="list-style-type: none"> ▪ Load policy: the policy for writing data to the destination table. Valid values: Append Data and Overwrite Data. ▪ Load Parameters: the parameters for connecting to MaxCompute. Example: <pre> {"accessid":"<yourAccessKeyId>","accessKey":"* <yourAccessKeySecret>","odpsServer":"*****", "tunnelServer":"*****","accountType":"aliyun","project":"transfer_project"} </pre> ▪ Alibaba Cloud Account: the Alibaba Cloud account that is authorized to write data, for example, ALIYUN\$garuda_data@aliyun.com.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields and perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.

Parameter	Description
Mapping	The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping .

8. Click **OK**.

Copy an ADB for MySQL2.0 output component

1. Right-click an ADB for MySQL2.0 output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ADB for MySQL2.0 output component

1. Right-click an ADB for MySQL2.0 output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.15. Manage ADB for MySQL3.0 output components

This topic describes how to configure, copy, and delete an ADB for MySQL3.0 output component.

Context

An ADB for MySQL3.0 output component is used to write data processed by Dataphin to an AnalyticDB for MySQL 3.0 database for data consumption.

Configure an ADB for MySQL3.0 output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an ADB for MySQL3.0 component to the pipeline canvas on the left.
6. Right-click the ADB for MySQL3.0 component and select **Configure Attributes**.
7. In the **AnalyticDB for MySQL 3.0 Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source.</p>
Table	The destination table.
Loading Policy	<p>The policy for writing data to the destination table. Valid values:</p> <ul style="list-style-type: none"> ◦ Overwrite Data: uses the source data to overwrite the existing data in the destination table. ◦ Append Data: appends the source data to the destination table, without modifying the existing data in the destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> ◦ Enter Preparation Statement: the SQL statements to execute before data output. ◦ Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields to perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click OK.

Copy an ADB for MySQL3.0 output component

1. Right-click an ADB for MySQL3.0 output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ADB for MySQL3.0 output component

1. Right-click an ADB for MySQL3.0 output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.2.9.16. Manage ADB for PostgreSQL output components

This topic describes how to configure, copy, and delete an ADB for PostgreSQL output component.

Context

An ADB for PostgreSQL output component is used to write data processed by Dataphin to an AnalyticDB for PostgreSQL database for data consumption.

Configure an ADB for PostgreSQL output component

1. [Log on to the Dataphin console](#).
2. Perform the following steps to create an offline migration pipeline:
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. In the top navigation bar, move the pointer over **Develop** and select **Integrated**. The **Integrated** page appears.
 - iv. On the **Integrated** page, move the pointer over the  icon next to **Script** and select **Offline Single Pipeline**.

v. In the **Create Pipeline Development Script** dialog box, set the parameters as required.

Parameter	Description
Pipeline Name	The name of the offline migration pipeline to be created.
Schedule Type	The scheduling type of the task corresponding to the offline migration pipeline. Valid values: <ul style="list-style-type: none"> ▪ Recurring Node: a task that is run on a specified schedule. ▪ One-Time Node: a task that has no dependencies and is triggered after you click Run.
Description	The description of the offline migration pipeline.
Select Directory	The folder where the offline migration pipeline resides.

vi. Click **OK**.

3. On the configuration tab of the pipeline, click **Component Library** in the upper-right corner.
4. Click the  icon before **Output**.
5. Drag an ADB for PostgreSQL component to the pipeline canvas on the left.
6. Right-click the ADB for PostgreSQL component and select **Configure Attributes**.
7. In the **AnalyticDB for PostgreSQL Output Configuration** dialog box, set the parameters as prompted.

Parameter	Description
Step Name	The name of the component. Enter a name based on the scenario.

Parameter	Description
Data Source	<p>The data source of the component. Select a data source that has been configured in Dataphin. The data source must meet the following requirements:</p> <ul style="list-style-type: none"> ◦ The data source is of the same type as the output component. ◦ The account that you use to configure the component has the write-through permission on the data source. If your account does not have the required permission, apply for permissions on the data source. For more information, see Apply for permissions on a data source. <p>You can also click the  icon next to Data Source to go to the Data Sources page and create a data source. For more information, see Create a MySQL data source.</p>
Table	The destination table.
Parse Solution	<p>Optional. The operations before and after data output. To specify operations before or after data output, click Enter Preparation Statement or Enter Completion Statement next to Parse Solution and enter SQL statements.</p> <ul style="list-style-type: none"> ◦ Enter Preparation Statement: the SQL statements to execute before data output. ◦ Enter Completion Statement: the SQL statements to execute after data output.
Input Fields	The input fields based on the upstream input.
Output Fields	<p>The output fields to be generated. You can click Manage Fields to perform the following operations in the dialog box that appears:</p> <ul style="list-style-type: none"> ◦ Select one or more fields in the Selected Input Fields section and click the  icon to move the fields to the Unselected Input Fields section. ◦ Select one or more fields in the Unselected Input Fields section and click the  icon to move the fields to the Selected Input Fields section.
Mapping	<p>The mapping between the input and output fields. To specify mappings one by one, click a field in the input field list and a field in the output field list. To specify multiple mappings at a time, click Quick Mapping and select Position-based Mapping or Name-based Mapping.</p>

8. Click **OK**.

Copy an ADB for PostgreSQL output component

1. Right-click an ADB for PostgreSQL output component and select **Copy**.
2. Right-click a blank area on the pipeline canvas and select **Paste**.

Delete an ADB for PostgreSQL output component

1. Right-click an ADB for PostgreSQL output component and select **Delete**.
2. In the message that appears, click **OK**.

9.8.3. Data synchronization

9.8.3.1. Create a folder for storing sync tasks

You can use different folders to manage different types of sync tasks. This topic describes how to create a folder for storing sync tasks and the operations that you can perform on created folders.

Procedure

1. Log on to the Dataphin console.
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The Sync Tasks section appears.
6. In the **Sync Tasks** section, click the  icon next to **Sync Tasks**.
7. In the **Create Folder** dialog box, set the **Name** and **Select Directory** parameters.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	To rename a folder, perform the following steps: <ol style="list-style-type: none">i. In the Sync Tasks section, move the pointer over the  icon next to the folder that you want to rename and select Rename.ii. In the field that appears, enter a new name.iii. Press the Enter key.
Move a folder	To move a folder, perform the following steps: <ol style="list-style-type: none">i. In the Sync Tasks section, move the pointer over the  icon next to the folder that you want to move and select Move.ii. In the Move Folder dialog box, set the Select Directory parameter to the destination directory.iii. Click OK.
Delete a folder	To delete a folder, perform the following steps: <ol style="list-style-type: none">i. In the Sync Tasks section, move the pointer over the  icon next to the folder that you want to delete.ii. Select Delete. <div style="background-color: #e0f2f7; padding: 5px;"><p> Note You can delete only the folders that do not contain subfolders or items.</p></div>

9.8.3.2. Create a destination table for data synchronization

A destination table for data synchronization is a table that Dataphin uses to process business data. It is in the computing engine bound to the project to which the business belongs. This topic describes how to create a destination table for data synchronization.

Context

Dataphin allows you to create a destination table for data synchronization by creating an ad hoc query task or a batch processing task.

Create a destination table by creating an ad hoc query task

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
5. Create a folder for storing ad hoc query tasks.
 - i. In the **Ad Hoc Query** section, click the  icon next to **Ad Hoc Query**.
 - ii. In the **Create Folder** dialog box, set the **Name** and **Select Directory** parameters.
 - iii. Click **OK**.
6. Create an ad hoc query task.
 - i. In the **Ad Hoc Query** section, click the  icon next to **Ad Hoc Query**.
 - ii. In the **Create Item** dialog box, set the **Name**, **Description**, and **Select Directory** parameters.
 - iii. Click **OK**. The **Code Editor** tab appears.
7. Write the code of the ad hoc query task on the **Code Editor** tab.

- i. Write SQL statements to create a destination table. In this example, write the following SQL statements:

```
-- Table<store_sales (23 cols) partition=ss_sold_date_sk>
drop table if exists store_sales;
create table if not exists store_sales(
  ss_sold_date_sk bigint
, ss_sold_time_sk bigint
, ss_item_sk bigint
, ss_customer_sk bigint
, ss_cdemo_sk bigint
, ss_hdemo_sk bigint
, ss_addr_sk bigint
, ss_store_sk bigint
, ss_promo_sk bigint
, ss_ticket_number bigint
, ss_quantity int
, ss_wholesale_cost double
, ss_list_price double
, ss_sales_price double
, ss_ext_discount_amt double
, ss_ext_sales_price double
, ss_ext_wholesale_cost double
, ss_ext_list_price double
, ss_ext_tax double
, ss_coupon_amt double
, ss_net_paid double
, ss_net_paid_inc_tax double
, ss_net_profit double
)
partitioned by (ds string)
```

- ii. Click **Precompile** in the upper-right corner of the Code Editor tab to check whether the SQL statement complies with the standard. If the SQL statements do not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statements so that they comply with the standard.
- iii. Click **Run** in the upper-right corner of the Code Editor tab to execute the SQL statements. Then, check whether the SQL statements are executed based on the information on the Console tab in the lower part of the page.

8. If the SQL statements are executed, click the  icon in the upper-right corner of the Code Editor tab.

Create a destination table by creating a batch processing task

1. On the Dataphin homepage, click **R&D** in the top navigation bar.
2. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
3. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
4. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The Batch Processing Tasks section appears.
5. Create a folder for storing batch processing tasks.
 - i. In the **Batch Processing Tasks** section, click the  icon next to **Batch Processing Tasks**.
 - ii. In the **Create Folder** dialog box, set the **Name** and **Select Directory** parameters.

- iii. Click **OK**.
6. Create a batch processing task of the `MAX_COMPUTE_SQL` type.
 - i. In the **Batch Processing Tasks** section, click the  icon next to **Batch Processing Tasks** and select `MAX_COMPUTE_SQL`.
 - ii. In the **Create Item** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the batch processing task, for example, SQL.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- iii. Click **OK**. The Code Editor tab appears.
7. Write the code of the batch processing task on the Code Editor tab.
 - i. Write SQL statements to create a destination table. In this example, write the following SQL statements:

```
-- Table<customer (18 cols)>
drop table if exists customer;
create table if not exists customer(
  c_customer_sk bigint
, c_customer_id string
, c_current_demo_sk bigint
, c_current_hdemo_sk bigint
, c_current_addr_sk bigint
, c_first_ship_to_date_sk bigint
, c_first_sales_date_sk bigint
, c_salutation string
, c_first_name string
, c_last_name string
, c_preferred_cust_flag string
, c_birth_day int
, c_birth_month int
, c_birth_year int
, c_birth_country string
, c_login string
, c_email_address string
, c_last_review_date_sk bigint
)
;
```

- ii. Click **Precompile** in the upper-right corner of the Code Editor tab to check whether the SQL statement complies with the standard. If the SQL statements do not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statements so that they comply with the standard.
- iii. Click **Run** in the upper-right corner of the Code Editor tab to execute the SQL statements. Then, check whether the SQL statements are executed based on the information on the Console tab in the lower part of the page.

8. (Optional) Configure the scheduling policy.
 - If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
9. Save, submit, and publish the batch processing task.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.
 - ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
 - iii. (Optional) Publish the batch processing task.
 - If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.8.3.3. Create a sync task

This topic describes how to create a sync task for synchronizing data from a business data source to a destination table in Dataphin.

Prerequisites

Destination tables for data synchronization are created. For more information, see [Create a destination table for data synchronization](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The Sync Tasks section appears.
6. Create a folder for storing sync tasks.
 - i. In the **Sync Tasks** section, click the  icon next to **Sync Tasks**.
 - ii. In the **Create Folder** dialog box, set the **Name** and **Select Directory** parameters.
 - iii. Click **OK**.
7. In the **Sync Tasks** section, click the  icon next to **Sync Tasks**.
8. In the **Create Item** dialog box, set the parameters as required.

Create File
✕

* Name

* Schedule Type Recurring Node One-Time Node

Description

Select Directory

Parameter	Description
Name	The name of the sync task.
Schedule Type	The scheduling type of the task. Valid values: <ul style="list-style-type: none"> ◦ Recurring Node: If you select this option, you must configure the scheduling policy. ◦ One-Time Node: If you select this option, you do not need to configure the scheduling policy.
Description	The description of the task.
Select Directory	The directory for storing the task.

9. Click OK. The sync task is created and is in the **Draft** state.

The following table describes the operations that you can perform on sync tasks in the **Draft** state.

Operation	Description
Modify a sync task	To modify a sync task, perform the following steps: <ol style="list-style-type: none"> i. In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to modify and select Change. ii. On the configuration tab of the sync task, modify the task settings. For more information, see Configure a sync task.
Rename a sync task	To rename a sync task, perform the following steps: <ol style="list-style-type: none"> i. In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.

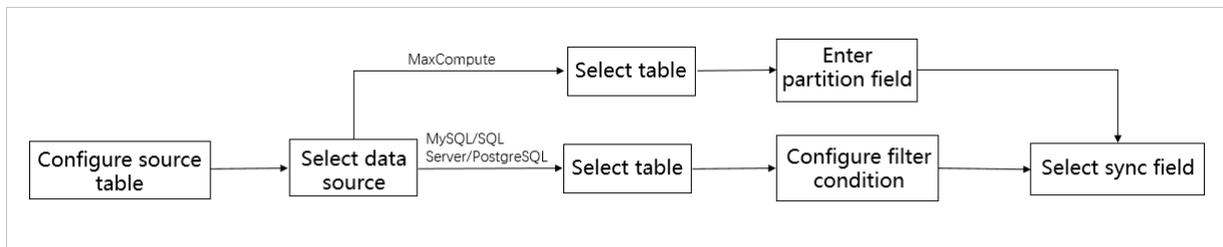
Operation	Description
Move a sync task	<p>To move a sync task, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination directory. iii. Click OK.
Delete a sync task	<p>To delete a sync task, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to delete and select Delete. ii. In the Tip dialog box, enter your comments. iii. Click OK.

9.8.3.4. Configure a sync task

You can configure a sync task to synchronize data from a source table to a destination table. This topic describes how to configure a sync task.

Configuration process

The following figure shows how to configure a sync task.



The following table describes the data sources that Dataphin supports for both the source and destination tables.

Data storage type	Data source
Relational database	MySQL, Vertica, Oracle, SQL Server, PostgreSQL, and PolarDB-X
Analytical database	AnalyticDB, AnalyticDB for MySQL V3.0, and AnalyticDB for PostgreSQL
Alibaba Cloud big data warehouse	MaxCompute
Application database	LogHub
Open-source big data warehouse	Hive and HBase

Data storage type	Data source
Unstructured data storage	FTP, HDFS, and Elasticsearch
NoSQL database	MongoDB

 **Note** If you select HBase, HBase 0.94.x and HBase 1.1.x are supported.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Sync Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode.
You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The Sync Tasks section appears.
3. In the **Sync Tasks** section, click a created sync task. On the tab that appears, set the parameters as required.
 - i. Set the parameters in the **Source** section. This section displays parameters based on the data source of the source table that you select. For more information about the parameters in this section, see [Data sources](#).
 - ii. Set the parameters in the **Target** section. This section displays parameters based on the data source of the destination table that you select. For more information about the parameters in this section, see [Data sources](#).
 - iii. Set the parameters in the **Source Fields** section. After you select a source table in the **Source** section, the fields of the source table appear in the **Source Fields** section. You can click the  icon in the **Actions** column to remove a field.
If you need to add a field that you have removed, click **Create Field**. In the dialog box that appears, check the **Field and Description** and **Data Type** parameters of the field, select the field, and then click **Add**.
 - iv. Set the parameters in the **Target Fields** section. After you select a destination table in the **Target** section, the fields of the destination table appear in the **Target Fields** section. You can click the  icon in the **Actions** column to remove a field.
If you need to add a field that you have removed, click **Create Field**. In the dialog box that appears, check the **Field and Description** and **Data Type** parameters of the field, select the field, and then click **Add**.

- v. Match the source and destination fields.
 - If each source field has the same name as a destination field and the number of source fields equals that of destination fields, the system automatically matches the source and destination fields.
 - If the source and destination fields have different names, you must manually match the source and destination fields. Remove the destination fields whose names are different from the source fields. Move the pointer over an empty row in the destination field list and select a field from the list that appears.
4. Set required limits for the sync task in the **Sync Limits** section. The parameters in this section are used to control the concurrency and fault tolerance. In most cases, you can use the default settings.

Parameter	Description
Speed Limit	The maximum data transmission rate during synchronization. Default value: 1MB/s. Dataphin will try to reach but will not exceed this rate. This parameter determines the size of scheduled resources when the sync task is running. The higher the speed limit is, the more resources are scheduled.
Concurrent Tasks	The maximum number of concurrent sync tasks.
Error Threshold	The maximum number of errors that are allowed during synchronization before the sync task is terminated. The default value is 0, which indicates that no errors are allowed.

5. Configure the scheduling policy for the sync task. For more information, see [Configure a scheduling policy](#).
6. Save, submit, and then publish the sync task.
- i. Click the  icon in the upper-right corner to save the sync task.
 - ii. Click the  icon in the upper-right corner to submit the sync task.
 - iii. (Optional) Publish the sync task.
 - If the current project is in Dev mode, publish the sync task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the sync task.

The following table describes the operations that you can perform on a sync task in the **Submitted** state.

Operation	Description
-----------	-------------

Operation	Description
Modify a sync task	<p>To modify a sync task, perform the following steps:</p> <ol style="list-style-type: none"> In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to modify and select Change. On the tab that appears, modify the configuration of the sync task. For more information, see Configure a sync task. <p>You can modify a sync task only when it is locked by you or unlocked. If the task is locked by another user, the  icon appears in the upper-right corner. You can click the  icon to steal the lock. After the lock is stolen, the  icon appears, and you can modify the sync task.</p> <div style="background-color: #e1f5fe; padding: 10px; border: 1px solid #cfcfcf;"> <p> Note After you modify a sync task in the Submitted state and save the modification, the status of the sync task changes to Developing.</p> </div>
Rename a sync task	<p>To rename a sync task, perform the following steps:</p> <ol style="list-style-type: none"> In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to rename and select Rename. In the field that appears, enter a new name and press Enter.
Move a sync task	<p>To move a sync task, perform the following steps:</p> <ol style="list-style-type: none"> In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to move and select Move. In the Move File dialog box, set the Select Directory parameter to the destination directory and click OK.
Unpublish a sync task	<p>To unpublish a sync task, perform the following steps:</p> <ul style="list-style-type: none"> ▪ In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to unpublish and select Unpublish. ▪ In the Tip dialog box, enter your comments and click OK.
Unpublish and delete a sync task	<p>To unpublish and delete a sync task, perform the following steps:</p> <ul style="list-style-type: none"> ▪ In the Sync Tasks section, move the pointer over the  icon next to the sync task that you want to unpublish and delete and select Unpublish and Delete. ▪ In the Tip dialog box, enter your comments and click OK.

9.8.3.5. Configure a scheduling policy

Dataphin allows you to configure the scheduling rules and dependencies of nodes to make sure that tasks can be properly scheduled. This topic describes how to configure a scheduling policy for a sync task.

Prerequisites

Sync tasks are configured. For more information, see [Configure a sync task](#).

Context

- Dataphin allows you to configure scheduling policies only for recurring tasks.
- A dependency is a semantic connection between two or more nodes. The status of an upstream node affects the running of its downstream nodes.
- After you configure dependencies for an upstream node and its downstream nodes, the downstream nodes can be run only after the upstream node is run. Before the system runs a node, the system checks the scheduling time that is configured for the node and determines whether to run the node.
- If a scheduling configuration is submitted before the specified scheduling time, the configuration takes effect after the specified scheduling time. If a dependency is configured after the specified scheduling time, the dependency takes effect for the instances that are generated the next day.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. Go to the **Sync Tasks** section.
 - i. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - ii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iii. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The Sync Tasks section appears.
4. In the **Sync Tasks** section, click a configured sync task.
5. On the tab that appears, click **Scheduling Configuration**.
6. In the **Scheduling Configuration** pane, set the parameters as required.
 - i. Set the parameters in the **Basic Information** section. Dataphin automatically generates the node name, node ID, node type, and owner. You cannot modify these parameters.

Parameter	Description
Description	The description of the scheduling policy.
Priority	The priority based on which the sync task is scheduled. Valid values: <ul style="list-style-type: none">▪ Lowest Priority▪ Low Priority▪ Medium Priority▪ High Priority▪ Highest Priority
Parameters	The specified values for the parameters that are used in the code of the task. Dataphin allows you to specify the parameters that are used in the code of a task. The specified values are used when the task is run. You can click Parameters and Descriptions to know how to configure the parameters.

ii. Set the parameters in the **Scheduling Configuration** section.

Parameter	Description
Schedule Mode	<p>The scheduling mode of the task. Valid values:</p> <ul style="list-style-type: none">▪ Normal: runs the task based on the specified recurrence. By default, this option is selected.▪ Dry-run: runs the task based on the specified recurrence. However, the scheduling system does not run the task but returns a success response.▪ Pause Scheduling: runs the task based on the specified recurrence. However, the scheduling system does not run the task but returns a failure response. You can select this check box if you need to suspend a task and run it later.

Parameter	Description
<p>Recurrence</p>	<p>The recurrence of the task. Valid values:</p> <ul style="list-style-type: none"> ▪ Day: automatically runs the task once per day. When you create a recurring task, the task is set to run at 00:00 every day by default. You can also click the  icon to specify a time for the task to be run as needed. ▪ Week: automatically runs the task at a specified time point on specified days of each week. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not actually run the instance or consume resources but directly returns a success response. <p>Assume that you set Recurrence to Week and specify that the task is run every Monday and Tuesday. The scheduling system generates and runs instances every Monday and Tuesday. Every Wednesday, Thursday, Friday, Saturday, and Sunday, the scheduling system generates instances and returns success responses without running the instances.</p> <ul style="list-style-type: none"> ▪ Month: automatically runs the task at a specified time point on specified days of each month. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not actually run the instance or consume resources but directly returns a success response. <p>Assume that you set Recurrence to Month and specify that the task is run on the seventh day of each month. The scheduling system generates and runs an instance on the seventh day of each month. On the other days, the scheduling system generates instances and returns success responses without running the instances.</p> <ul style="list-style-type: none"> ▪ Hour: automatically runs the task at a specified interval during a specified time period or at specified time points every day. The scheduling system automatically generates instances for the task and runs the instances at the specified interval or time points. <p>Assume that you set Recurrence to Hour, select Time Period, and set the Start, End, and Interval parameters to 00:00, 23:59, and 1, respectively. The scheduling system automatically generates instances for the task and runs an instance every hour.</p> <ul style="list-style-type: none"> ▪ Minute: automatically runs the task at a specified interval during a specified time period every day. <p>Assume that you set Recurrence to Minute and set the Start, End, and Interval parameters to 00:00, 23:59, and 05, respectively. The scheduling system automatically generates instances for the task and runs an instance every 5 minutes.</p>
<p>Depend on Previous Instance</p>	<p>Specifies whether the current task is run after the previous instance of another task or of the current task is run. If you select Depend on Previous Instance, you must further select Current Task or Select Task.</p> <ul style="list-style-type: none"> ▪ If you select Current Task, the current task is run after the previous instance of the current task is run. ▪ If you select Select Task, enter a keyword in the search box that appears to search for and select one or more tasks to depend on.

iii. Set the parameters in the **Dependency** section.

Parameter	Description
Upstream Dependency	<p>The upstream nodes on which the current node depends. To specify an upstream node, perform the following steps:</p> <ol style="list-style-type: none"> Click Create Upstream Dependency. In the Create Upstream Dependency dialog box, search for a node based on the output name. <div style="border: 1px solid #ccc; background-color: #e0f2f1; padding: 5px; margin: 10px 0;"> <p> Note Each node output name is globally unique in Dataphin.</p> </div> <ol style="list-style-type: none"> Click OK. <p>To remove a node from the upstream node list, click the  icon in the Actions column.</p>
Current Node	<p>The output name for the current node. You can set multiple output names for a node, which can be used to configure dependencies for other nodes. To set an output name, perform the following steps:</p> <ol style="list-style-type: none"> Click Add. In the Add Output Task Nodes for Current Task Node dialog box, enter an output name. Observe uniform rules when you set each output name for the current node. Set an output name in the format of Project name. Table name. This helps other users find this node when they configure the upstream dependency for their nodes. Click OK. <p>You can also perform the following operations on an existing output name:</p> <ul style="list-style-type: none"> ■ To delete an output name, click the  icon in the Actions column. ■ To view the downstream nodes after the current node is submitted or published, click the  icon in the Actions column.

7. Click **OK**.

8. Save, submit, and then publish the sync task.

- On the configuration tab of the sync task, click the  icon in the upper-right corner to save the sync task.
- On the configuration tab of the sync task, click the  icon in the upper-right corner to submit the sync task.
- (Optional) Publish the sync task.
 - If the current project is in Dev mode, publish the sync task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the sync task. The task can be scheduled in the production environment after you submit it.

9.8.3.6. Run a one-time sync task

This topic describes how to run a one-time task.

Prerequisites

- One-time tasks are created. For more information, see [Create a sync task](#).
- One-time tasks are configured. For more information, see [Configure a sync task](#).

Context

Description on running of sync tasks:

- You do not need to run recurring tasks. Dataphin automatically runs recurring tasks based on their scheduling settings.
- One-time tasks run only after you trigger them manually.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click **Scheduling** in the top navigation bar.
5. On the **Scheduling** page, move the pointer over the left-side navigation submenu and click the  icon. The One-time Tasks section appears.
6. Run a one-time task.
 - i. In the **One-Time Tasks** section, click the one-time task that you want to run.
 - ii. On the page that appears, click **Run** in the upper-right corner.
 - iii. In the **Run** dialog box, set the **Instance Name** and **Data Timestamp** parameters based on your business needs. The default value of **Data Timestamp** is the date of the day before.
 - iv. Click **OK**.

9.8.3.7. Verify a sync task

This topic describes how to verify a sync task.

Prerequisites

Sync tasks are created and run. For more information about how to run a one-time task, see [Run a one-time sync task](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
5. Create a folder for storing ad hoc query tasks.
 - i. In the **Ad Hoc Query** section, click the  icon next to **Ad Hoc Query**.
 - ii. In the **Create Folder** dialog box, enter a folder name and select a directory.

- iii. Click **OK**.
6. Create an ad hoc query task.
 - i. In the **Ad Hoc Query** section, click the  icon next to **Ad Hoc Query**.
 - ii. In the **Create Item** dialog box, set the **Name**, **Description**, and **Select Directory** parameters.
 - iii. Click **OK**. The Code Editor tab appears.
7. Write the code of the ad hoc query task on the Code Editor tab.
 - i. Enter the following SQL statement on the Code Editor tab:

```
select * from Destination table name where ds='Data timestamp';
```
 - ii. Click **Precompile** in the upper-right corner of the Code Editor tab to check whether the SQL statement complies with the standard. If the SQL statement does not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statement so that it complies with the standard.
 - iii. Click **Run** in the upper-right corner of the Code Editor tab to execute the SQL statement. Then, check whether the sync task is running properly based on the result returned by the ad hoc query task.

9.9. Data development

9.9.1. Standards and modeling

9.9.1.1. Overview

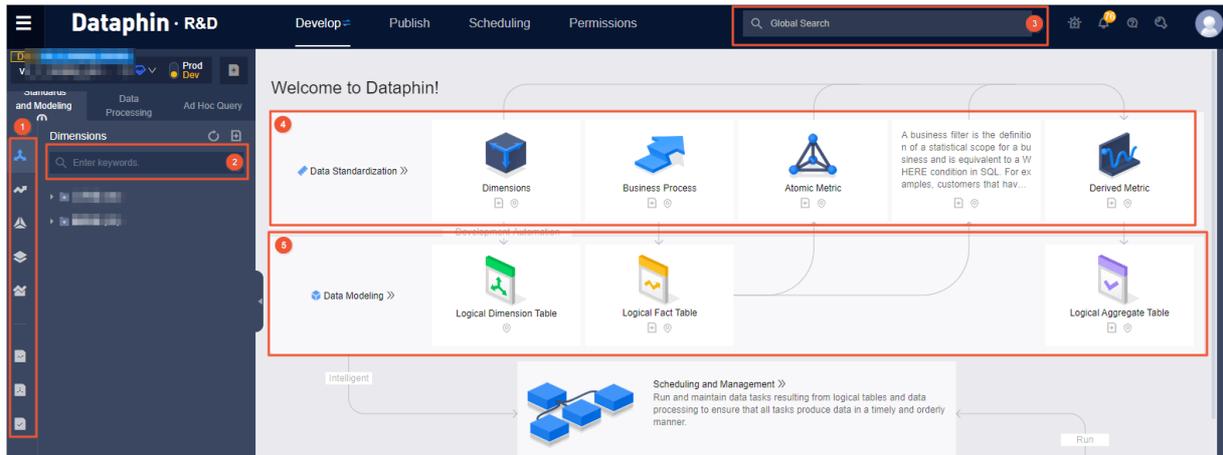
Dataphin provides the data standardization and data modeling features for building data warehouses. These features allow you to build data models in a systematic manner.

Go to the Standards and Modeling tab

To go to the **Standards and Modeling** tab, perform the following steps:

1. [Log on to the Dataphin console](#).
2. Go to the **Standards and Modeling** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.

On the **Standards and Modeling** tab, you can create data standardization objects and logical tables.



No.	Description
1	The left-side navigation submenu. Move the pointer over this submenu and click an item as needed. In the section that appears, you can create a corresponding data standardization object or logical table.
2	The local search box. Enter a keyword in the search box to search for data standardization objects or logical tables.
3	The global search box. Enter a keyword in the search box to search for objects in Dataphin. To search for a data distilling object, you must have activated the data distilling feature.
4	The Data Standardization section. This section displays the types of data standardization objects. Click the  icon below each type to create a corresponding data standardization object.
5	The Data Modeling section. This section displays the types of logical tables that are supported for data modeling. Click the  icon below each type to create a corresponding logical table.

Prerequisites

- External data is imported to the data sources that are configured for the current project. For more information, see [Data ingestion](#).
- The current project is in Basic or Dev mode. After you go to the Develop page, select a project in the upper-left corner first. Dataphin allows you to use the data standardization and data modeling features for projects in Basic or Dev-Prod mode. For more information, see [Create projects](#) and [Create business units](#).
- The current project is bound to a business unit. Otherwise, the data standardization and data modeling features are unavailable. For more information about how to bind a project to a business unit, see [Create projects](#).
- If the current project is in Basic mode, the project is bound to a business unit in Basic mode. If the project in Basic mode is bound to a business unit in Prod mode, you cannot perform data standardization and data modeling. You can only use the data processing and ad hoc query features.
- If the current project is in Dev-Prod mode, the project mode switch is switched to Dev, so that you can create, modify, or delete objects. After you submit a created or modified object, publish the object. After the object is published, it can be automatically scheduled in the corresponding project in Prod mode.

9.9.1.2. Data standardization: Dimensions

9.9.1.2.1. Create a dimension

A dimension reflects a perspective from which you view an object. You may need to view an object from multiple aspects or at multiple levels. To this end, Dataphin allows you to define dimensions for data standardization. This topic describes how to create a dimension.

Prerequisites

Data domains are created. For more information, see [Create a data domain](#).

Context

A dimension is a statistical object. It is an entity that exists, for example, the time, region, and product dimensions. By creating dimensions, you can standardize your business entities or master data during architectural design to guarantee that they are unique.

Procedure

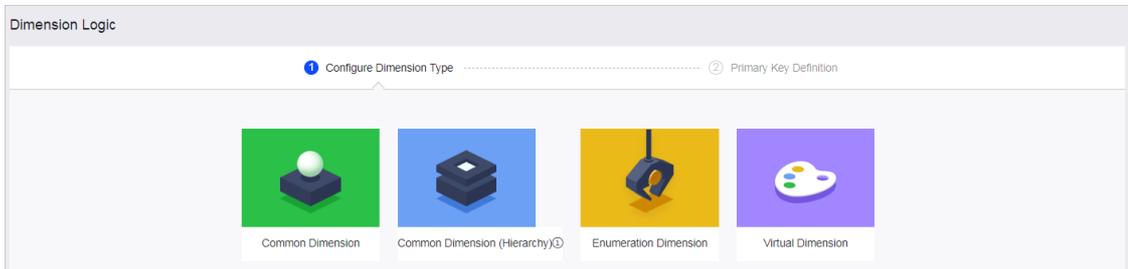
1. [Log on to the Dataphin console](#).
2. Go to the **Dimensions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Dimensions** section appears.
3. Go to the **Create Dimension** tab by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Dimensions**.
 - In the **Dimensions** section, click the  icon next to **Dimensions**.
 - In the right-side workspace of the **Develop** page, click the  icon below **Dimensions**.
 - In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Click **Create Dimension**.
4. On the **Create Dimension** tab, set the parameters as required.

i. Set the parameters in the **Basic Dimension Information** section.

Parameter	Description
Dimension Name	The name of the dimension. The name can contain letters, digits, and underscores (_). By default, the name starts with the dim_ prefix.
Dimension Display Name	The display name of the dimension. The display name can contain letters, digits, underscores (_), and hyphens (-).
Dimension Description	The description of the dimension to be created.
Data Domain	The data domain to which the dimension belongs. If no data domain is available, create one first.

ii. Set the parameters in the **Dimension Logic** section.

The dimension logic defines the scope of the dimension and determines the logical characteristics of the dimension. The dimension logic ensures that the dimension is true and unique. The following table describes the four dimension types provided by Dataphin to meet different needs.



Dimension type	Description
Common Dimension	To create a common dimension, enter the primary key information. Set the parent dimension as needed. After you create and publish a common dimension, Dataphin automatically generates a logical dimension table.
Common Dimension (Hierarchy)	To create a common dimension by hierarchy, enter the primary key information and define the dimension hierarchy and logic. You can analyze data from multiple dimensions in a stable hierarchy. After you create and publish a common dimension by hierarchy, Dataphin automatically generates a set of logical dimension tables by hierarchy.
Enumeration Dimension	To define an enumeration dimension, enter enumeration codes and values.
Virtual Dimension	To create a virtual dimension, enter the primary key information. Dataphin allows you to define virtual dimensions that have no business entities and cannot be specified by a fixed data scope in a source table. A virtual dimension can also be used in data standardization and data modeling.

- If you select **Common Dimension**, set the parameters as described in the following table.

Parameter	Description
Primary Key Name	The name of the primary key for the dimension. The name can contain letters, digits, and underscores (_).
Primary Key Display Name	The display name of the primary key for the dimension. The display name can contain letters, digits, underscores (_), and hyphens (-).
Primary Key Type	The data type of the primary key for the dimension. Valid values: STRING , BIGINT , DOUBLE , DATETIME , and DECIMAL .
Primary Key Computing Logic	<p>The computing logic of the primary key for the dimension. To prevent maintenance exceptions or data errors, configure the time-based partitioning conditions in the code.</p> <ol style="list-style-type: none"> Click Example to view the description about the computing logic of the primary key for the dimension and the sample SQL statement. <pre style="background-color: #f0f0f0; padding: 5px;">select province from dataphin_test where ds='\${bizdate}';</pre> <ol style="list-style-type: none"> After you enter an SQL statement, click Code Check to verify whether the SQL statement you entered is valid.
Parent Dimension	The parent dimension of the dimension. Valid values: Yes and No . If you select Yes , select a parent dimension from the drop-down list.

- If you select **Common Dimension (Hierarchy)**, perform the following steps to set the parameters:

a. Set the parameters in the **Primary Key and Source Logic Definition** step and click **Next**.

Parameter	Description
Primary Key Name	The name of the primary key for the dimension. The name can contain letters, digits, and underscores (_).
Primary Key Display Name	The display name of the primary key for the dimension. The display name can contain letters, digits, underscores (_), and hyphens (-).
Primary Key Type	The data type of the primary key for the dimension. Valid values: STRING , BIGINT , DOUBLE , DATETIME , and DECIMAL .
Primary Table Name	The name of the main source table for the dimension. This table is also the destination table for a sync task. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note We recommend that you select a table from the production environment.</p> </div>

b. Set the parameters in the **Hierarchy Definition** step.

Parameter	Description
Child Field	The child field in the hierarchy.
Parent Field	The parent field in the hierarchy.
Name Field	The field of the dimension name in the hierarchy.
Number of Levels	The number of levels in the hierarchy. A maximum of nine levels are supported.
Generate Leaf Dimension	Specifies whether to generate a leaf dimension. Valid values: <ul style="list-style-type: none"> ■ Yes: Dataphin automatically generates a leaf dimension. ■ No: Dataphin generates a leaf dimension only if you have defined the computing logic for the leaf dimension.
Root-level Dimension Definition	The judgment condition for the root-level dimension based on the field in the main source table. For example, you can enter <code>is_parent='ture'</code> , where <code>parent</code> indicates the field in the main source table.
Time-based Partitioning	The judgment condition for the data timestamp and statistical period based on the field in the main source table. For example, you can enter <code>ds=\${bizdate}</code> .

- If you select **Enumeration Dimension**, set the parameters as instructed.

In the left-side field of the **Configure Enumeration Information** section, enter enumeration codes and values. Separate each enumeration code and its value with a comma (,).

F,female
M,male

In the right-side field, Dataphin automatically parses the enumeration codes and values that you entered in the **Code** and **Value** columns, respectively.

- If you select **Virtual Dimension**, set the parameters as described in the following table.

Parameter	Description
Primary Key Name	The name of the primary key for the dimension. The name can contain letters, digits, and underscores (_).
Primary Key Display Name	The display name of the primary key for the dimension. The display name can contain letters, digits, underscores (_), and hyphens (-).
Primary Key Type	The data type of the primary key for the dimension. Valid values: STRING , BIGINT , DOUBLE , DATETIME , and DECIMAL .

Note If a virtual dimension is associated with a logical dimension table or a logical fact table, you can create derived metrics based on this virtual dimension. The value of the virtual dimension used to compose the statistic granularity is extracted from the associated field.

5. After you set the parameters on the **Create Dimension** tab, click the  icon in the upper-right corner to save the dimension.
6. Submit the dimension.
 - i. On the **Create Dimension** tab, click the  icon in the upper-right corner to submit the dimension.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
 - After you submit a common dimension, Dataphin automatically generates the corresponding logical dimension table for the dimension.
 - After you submit a common dimension by hierarchy, Dataphin generates a set of logical dimension tables by hierarchy for the dimension.

7. (Optional) Publish the dimension.
 - o If the current project is in Dev mode, publish the dimension to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the dimension after you submit it.

9.9.1.2.2. View the logical dimension table of a dimension

This topic describes how to view the logical dimension table that is generated for a dimension.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Dimensions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Dimensions** section appears.
3. View the logical dimension table of a dimension by using one of the following methods:
 - o In the **Dimensions** section, click the data domain to which the dimension for which you want to view the logical dimension table belongs and the type of the dimension. Click the dimension. On the **View Attributes** tab, move the pointer over the  icon and select **Logical Dimension Table**.
 - o In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Find the dimension for which you want to view the logical dimension table and click the  icon in the **Actions** column.
4. On the tab that appears, view the information about the logical dimension table of the dimension.

9.9.1.2.3. Modify a dimension

A dimension reflects a perspective from which you view an object. You may need to view an object from multiple aspects or at multiple levels. To this end, Dataphin allows you to define dimensions for data standardization. This topic describes how to modify a dimension.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Dimensions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Dimensions** section appears.
3. Go to the **Change Dimension** tab by using one of the following methods:
 - In the **Dimensions** section, click the data domain to which the dimension that you want to modify belongs and the type of the dimension. Click the dimension. On the **View Attributes** tab, click **Change**.
 - In the **Dimensions** section, click the data domain to which the dimension that you want to modify belongs and the type of the dimension. Move the pointer over the  icon next to the dimension and select **Change**.
 - In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Find the dimension that you want to modify and click the  icon in the **Actions** column.
4. (Optional)Steal the lock of the dimension.
 - If the dimension is locked by yourself, you do not need to steal the lock.
 - If the dimension is locked by another user, click the  icon in the upper-right corner of the **Change Dimension** tab to steal the lock.
5. On the **Change Dimension** tab, modify the parameters as needed. For more information, see [Create a dimension](#).
6. After you modify the parameters on the **Change Dimension** tab, click the  icon in the upper-right corner to save the dimension.
7. Submit the dimension.
 - i. On the **Change Dimension** tab, click the  icon in the upper-right corner to submit the dimension.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
8. (Optional)Publish the dimension.
 - If the current project is in **Dev** mode, publish the dimension to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the dimension after you submit it.

9.9.1.2.4. Unpublish and delete a dimension

This topic describes how to unpublish, unpublish and delete, and delete dimensions in different states.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Context

- A dimension may be in the following states:
 - After you create and save a dimension, it enters the **Draft** state.
 - After you submit a dimension, it enters the **Submitted** state.

- After you modify and save a dimension in the **Submitted** state, it enters the **Developing** state.
- After you unpublish a dimension in the **Submitted** state, it enters the **Draft** state.
- You can unpublish only the dimensions in the **Developing** or **Submitted** state.
- You can delete only the dimensions in the **Draft** state.
- You can unpublish and delete only the dimensions in the **Developing** or **Submitted** state.

Unpublish a dimension

 **Note** You can unpublish only a dimension for which no dependency is configured.

1. [Log on to the Dataphin console.](#)
2. Go to the **Dimensions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Dimensions** section appears.
3. (Optional)Steal the lock of the dimension to be unpublished.
 - If the dimension is locked by yourself, you do not need to steal the lock.
 - If the dimension is locked by another user, click the  icon in the upper-right corner of the **Change Dimension** tab to steal the lock. For more information about how to go to the **Change Dimension** tab, see [Modify a dimension.](#)
4. Open the **Tip** dialog box for unpublishing a dimension by using one of the following methods:
 - In the **Dimensions** section, click the data domain to which the dimension that you want to unpublish belongs and the type of the dimension. Click the dimension. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish**.
 - In the **Dimensions** section, click the data domain to which the dimension that you want to unpublish belongs and the type of the dimension. Move the pointer over the  icon next to the dimension and select **Unpublish**.
 - In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Find the dimension that you want to unpublish and choose  > **Unpublish** in the **Actions** column.
5. In the **Tip** dialog box, enter your comments.
6. Click **OK**.

Delete a dimension

1. (Optional)Steal the lock of the dimension to be deleted.
 - If the dimension is locked by yourself, you do not need to steal the lock.
 - If the dimension is locked by another user, click the  icon in the upper-right corner of the **Change Dimension** tab to steal the lock. For more information about how to go to the **Change Dimension** tab, see [Modify a dimension.](#)

2. Open the **Tip** dialog box for deleting a dimension by using one of the following methods:
 - In the **Dimensions** section, click the data domain to which the dimension that you want to delete belongs and the type of the dimension. Move the pointer over the  icon next to the dimension and select **Delete**.
 - In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Find the dimension that you want to delete and choose  > **Delete** in the **Actions** column.
3. In the **Tip** dialog box, enter your comments.
4. Click **OK**.

Unpublish and delete a dimension

 **Note** You can unpublish and delete only a dimension for which no dependency is configured.

1. (Optional) Steal the lock of the dimension to be unpublished and deleted.
 - If the dimension is locked by yourself, you do not need to steal the lock.
 - If the dimension is locked by another user, click the  icon in the upper-right corner of the **Change Dimension** tab to steal the lock. For more information about how to go to the **Change Dimension** tab, see [Modify a dimension](#).
2. Open the **Tip** dialog box for unpublishing and deleting a dimension by using one of the following methods:
 - In the **Dimensions** section, click the data domain to which the dimension that you want to unpublish and delete belongs and the type of the dimension. Move the pointer over the  icon next to the dimension and select **Unpublish and Delete**.
 - In the **Dimensions** section, click **Dimensions Object List** in the lower part. The **Dimensions** tab of the **Object Explorer** tab appears. Find the dimension that you want to unpublish and delete and choose  > **Unpublish and Delete** in the **Actions** column.
 - In the **Dimensions** section, click the data domain to which the dimension that you want to unpublish and delete belongs and the type of the dimension. Click the dimension. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish and Delete**.
3. In the **Tip** dialog box, enter your comments.
4. Click **OK**.

9.9.1.3. Data standardization: Business processes

9.9.1.3.1. Create a business process

A business process describes the events that are necessary for specific business. For example, a business process in commerce involves placing orders, paying, and refunding. By creating a business process, you can standardize the events in specific business during architectural design and make sure that they are unique. This topic describes how to create a business process.

Prerequisites

Data domains are created. For more information, see [Create a data domain](#).

Procedure

1. Log on to the Dataphin console.
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Processes section appears.
3. Open the **Create Business Process** dialog box by using one of the following methods:
 - o In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Business Processes**.
 - o In the Business Processes section, click the  icon next to **Business Processes**.
 - o In the right-side workspace of the Develop page, click the  icon below **Business Process**.
 - o In the Business Processes section, click **Business Processes Object List** in the lower part. The **Business Processes** tab of the **Object Explorer** tab appears. Click **Create Business Process**.
4. In the **Create Business Process** dialog box, set the parameters as required.

Parameter	Description
Data Domain	The data domain to which the business process to be created belongs.
Name	The name of the business process. The name can contain letters, digits, and underscores (_).
Display Name	The display name of the business process. The display name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the business process.

5. Submit the business process.
 - i. After you set the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the business process.
 - o If the current project is in Dev mode, publish the business process to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the business process after you submit it.

9.9.1.3.2. Modify a business process

A business process describes the events that are necessary for specific business. For example, a business process in commerce involves placing orders, paying, and refunding. This topic describes how to modify a business process.

Prerequisites

Business processes are created. For more information, see [Create a business process](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Processes section appears.
3. Open the **Change Business Process** dialog box by using one of the following methods:
 - o In the **Business Processes** section, click the data domain to which the business process that you want to modify belongs. Click the business process. On the View Attributes tab, click **Change**.
 - o In the **Business Processes** section, click the data domain to which the business process that you want to modify belongs. Move the pointer over the  icon next to the business process and select **Change**.
 - o In the **Business Processes** section, click **Business Processes Object List** in the lower part. The **Business Processes** tab of the **Object Explorer** tab appears. Find the business process that you want to modify and click the  icon in the **Actions** column.
4. In the **Change Business Process** dialog box, modify the parameters as needed. For more information, see [Create a business process](#).
5. Submit the business process.
 - i. After you modify the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the business process.
 - o If the current project is in Dev mode, publish the business process to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the business process after you submit it.

9.9.1.3.3. Clone a business process

This topic describes how to clone a business process.

Prerequisites

Business processes are created. For more information, see [Create a business process](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Processes section appears.
3. Open the **Create Business Process** dialog box by using one of the following methods:
 - o In the **Business Processes** section, click the data domain to which the business process that you want to clone belongs. Click the business process. On the **View Attributes** tab, move the pointer over the  icon and select **Clone**.
 - o In the **Business Processes** section, click **Business Processes Object List** in the lower part. The **Business Processes** tab of the **Object Explorer** tab appears. Find the business process that you want to clone and choose  > **Clone** in the **Actions** column.
4. In the **Create Business Process** dialog box, set the parameters as required. For more information, see [Create a business process](#).
5. Submit the business process.
 - i. After you set the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the business process.
 - o If the current project is in **Dev** mode, publish the business process to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - o If the current project is in **Basic** mode, you do not need to publish the business process after you submit it.

9.9.1.3.4. Create a logical table

This topic describes how to create a logical fact table based on a business process.

Prerequisites

Business processes are created. For more information, see [Create a business process](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Processes section appears.
3. Open the **Create Logical Fact Table** dialog box by using one of the following methods:
 - o In the **Business Processes** section, click the data domain to which the business process for which you

want to create a logical fact table belongs. Click the business process. On the View Attributes tab, move the pointer over the  icon and select **Create Logical Table**.

- In the **Business Processes** section, click **Business Processes Object List** in the lower part. The **Business Processes** tab of the **Object Explorer** tab appears. Find the business process for which you want to create a logical fact table and click the  icon in the **Actions** column.
4. In the **Create Logical Fact Table** dialog box, set the parameters as required. For more information, see [Create a logical fact table](#).
 5. Submit and publish the logical fact table that you create. For more information, see [Create a logical fact table](#).

9.9.1.3.5. View logical fact tables related to a business process

This topic describes how to view logical fact tables that are related to a business process.

Prerequisites

Business processes are created. For more information, see [Create a business process](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Business Processes** section appears.
3. In the **Business Processes** section, click **Business Processes Object List** in the lower part.
4. On the **Business Processes** tab of the **Object Explorer** tab, find the business process for which you want to view related logical fact tables and click the  icon in the **Actions** column.
5. In the **Related Logical Fact Tables** list, view the logical fact tables that are related to the business process.

9.9.1.3.6. Delete a business process

A business process describes the events that are necessary for specific business. For example, a business process in commerce involves placing orders, paying, and refunding. This topic describes how to delete a business process.

Prerequisites

Business processes are created. For more information, see [Create a business process](#).

Context

You can delete only the business processes that have no related logical fact tables.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Business Processes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Processes section appears.
3. Open the **Tip** dialog box for deleting a business process by using one of the following methods:
 - In the **Business Processes** section, click the data domain to which the business process that you want to delete belongs. Click the business process. On the **View Attributes** tab, move the pointer over the  icon and select **Delete**.
 - In the **Business Processes** section, click the data domain to which the business process that you want to delete belongs. Move the pointer over the  icon next to the business process and select **Delete**.
 - In the **Business Processes** section, click **Business Processes Object List** in the lower part. The **Business Processes** tab of the **Object Explorer** tab appears. Find the business process that you want to delete and choose  > **Delete** in the **Actions** column.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

9.9.1.4. Logical tables: Logical dimension tables

9.9.1.4.1. Edit model information

One logical dimension table corresponds to one dimension. A logical dimension table stores dimension attributes that describe facts. After you submit or publish a dimension, Dataphin automatically generates a logical dimension table. For the logical dimension table, Dataphin allows you to add attributes, create a dimension association, add child dimensions, and configure a logical table conversion task.

Usage notes

You can edit a logical dimension table by creating a dimension association, adding attributes, and adding child dimensions. Note the following points when you edit model information:

- You only need to store persistent attributes of a dimension in its corresponding logical dimension table.
- You do not need to define fields for child dimensions if these fields are defined in the published parent dimension. Dataphin can automatically reference such fields.

Create a dimension association

1. [Log on to the Dataphin console.](#)
2. Go to the **Logical Dimension Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.

- iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Logical Dimension Tables** section appears.
3. In the **Logical Dimension Tables** section, click the logical dimension table for which you want to create a dimension association. The configuration tab of the logical dimension table appears.
 4. On the configuration tab of the logical dimension table, click **Create Dimension Association** in the **Central Table** section.
 5. In the **Create Dimension Association** dialog box, set the parameters as required.

Section	Description
Associated Dimension	The dimension to be associated with the current logical dimension table.
Edit Association Logic	The logic for associating a field in the current logical dimension table with the specified dimension. Select a field from the drop-down list under Dimension Table Field to Be Associated .
Edit Dimension Role	The role of the specified dimension. In this section, you must specify the Role Name and Role Display Name parameters.
Default Value Settings	The value of the specified field when the field cannot be associated with the specified dimension. By default, the Default Value parameter is set to -110.

6. Click **OK**.

Add attributes

1. In the **Logical Dimension Tables** section, click the logical dimension table for which you want to add attributes. The configuration tab of the logical dimension table appears.
2. On the configuration tab of the logical dimension table, click **Add Attributes** in the **Central Table** section.
3. In the **Create Attribute** dialog box, set the parameters as required.
 - If you set the **Source Table** parameter to **Import Fields**, perform the following steps to add fields:
 - Click the  icon next to each desired field in the **Select New Fields** section to add the field to the **Create Field** section. Then, enter a display name for the field in the **Field Display Name** column. Click **Save and Verify**.
 - If you set the **Source Table** parameter to **Customize Fields**, enter SQL statements in the code editor to define fields. You can click **Example** in the upper part of the code editor to view the sample SQL statement. Click **Code Check** to verify the syntax of the SQL statements you entered. After the SQL statements pass the verification, click **Save and Verify**.

Create a child dimension

1. In the **Logical Dimension Tables** section, click the logical dimension table for which you want to create a child dimension. The configuration tab of the logical dimension table appears.
2. On the configuration tab of the logical dimension table, click **Add Child Dimension** in the **Central Table** section.
3. On the **Inherit from Dimension** tab, set the parameters as required.

Section	Parameter	Description
Basic Dimension Information	Dimension Name	The name of the child dimension.
	Dimension Display Name	The display name of the child dimension.
	Dimension Description	The description of the child dimension.
Dimension Logic	Primary Key Name	The name of the primary key of the child dimension.
	Primary Key Display Name	The display name of the primary key of the child dimension.
	Primary Key Type	The data type of the primary key of the child dimension. Click the <input type="checkbox"/> icon and select a data type from the drop-down list.
	Primary Key Computing Logic	The primary key computing logic that you want to define for the child dimension. Perform the following steps: <ol style="list-style-type: none"> i. Click Example to view the description about the primary key computing logic and the sample SQL statement. ii. In the code editor, enter SQL statements for defining the primary key computing logic. iii. Click Code Check to verify the syntax of the SQL statements you entered.
	Parent Dimension	The parent dimension of the child dimension to be created. By default, the dimension that corresponds to the current logical dimension table is selected as the parent dimension. You can also click the <input type="checkbox"/> icon and select a parent dimension from the drop-down list.

4. On the **Inherit from Dimension** tab, click the  icon in the upper-right corner to save the inherited dimension.
5. Submit the inherited dimension.
 - i. On the **Inherit from Dimension** tab, click the  icon in the upper-right corner to submit the inherited dimension.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. Publish the inherited dimension.
 - If the current project is in Dev mode, publish the inherited dimension to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the inherited dimension after you submit it.

9.9.1.4.2. Edit a central table

This topic describes how to edit a central table for a logical dimension table, such as adding attributes, creating dimension associations, and defining computing logic.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Add attributes

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Dimension Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Logical Dimension Tables** section appears.
3. In the **Logical Dimension Tables** section, click the logical dimension table to which you want to add attributes. The configuration tab of the logical dimension table appears.
4. (Optional)Steal the lock of the logical dimension table.
 - o If the logical dimension table is locked by yourself, you do not need to steal the lock.
 - o If the logical dimension table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
6. On the **Central Table Settings** tab, move the pointer over **Create Field** and select **Attribute**.
7. In the **Create Attribute** dialog box, set the parameters as required. For more information, see [Add attributes](#).
8. Save, submit, and then publish the logical dimension table.
 - i. After you add attributes, click the  icon in the upper-right corner of the configuration tab to save the attributes.
 - ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
 - v. (Optional)Publish the logical dimension table.
 - If the current project is in **Dev** mode, publish the logical dimension table to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the logical dimension table after you submit it.

Create a dimension association

1. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
2. On the **Central Table Settings** tab, move the pointer over **Create Field** and select **Dimension Association**.
3. In the **Create Dimension Association** dialog box, set the parameters as required. For more information, see [Create a dimension association](#).
4. Save, submit, and then publish the logical dimension table.
 - i. After you create a dimension association, click the  icon in the upper-right corner of the configuration tab to save the dimension association.
 - ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
 - v. (Optional) Publish the logical dimension table.
 - If the current project is in Dev mode, publish the logical dimension table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical dimension table after you submit it.

Define public computing logic

1. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
2. On the **Central Table Settings** tab, click **Define Public Computing Logic** next to **Create Field**.
3. In the **Define Public Computing Logic** dialog box, read the description on the right and click **Create Public Computing Logic** on the left.
4. In the dialog box that appears, enter a name for the public computing logic in the field and click **OK**. A code editor appears.
5. Perform the following steps to define the public computing logic:
 - i. Click **Example** to view the description about the public computing logic and the sample SQL statement.
 - ii. Enter SQL statements.
 - iii. Click **Code Check** to verify the SQL statements you entered.

If the SQL statements do not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statements so that they comply with the standard.
 - iv. After the SQL statements pass the verification, click **Save**.You can also perform the following steps to delete existing computing logic:
 - Select the computing logic to be deleted or click **Select All** in the lower part of the **Define Public Computing Logic** dialog box.
 - Click **Delete**.
 - In the **Delete Public Computing Logic** message, click **OK**.
 - Another **Delete Public Computing Logic** message appears. Click **OK**.
6. Click **Finish** in the lower-right corner of the **Define Public Computing Logic** dialog box.
7. Save, submit, and then publish the logical dimension table.

- i. After you define the computing logic, click the  icon in the upper-right corner of the configuration tab to save the computing logic.
- ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
- iii. In the dialog box that appears, enter your comments.
- iv. Click **OK**.
- v. (Optional) Publish the logical dimension table.
 - If the current project is in Dev mode, publish the logical dimension table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical dimension table after you submit it.

View computing logic

1. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
2. On the **Central Table Settings** tab, click **Group by Type** to view the primary key, attributes, associated dimensions, and system field of the logical dimension table. Click **Group by Source** to view the primary key, system field, and source logical tables of the logical dimension table.
3. On the **Group by Type** or **Group by Source** tab, find the field for which you want to view the computing logic and click the  icon in the **Computing Logic (All)** column.

You can use one of the following methods to filter fields:

- Click the  icon next to **Computing Logic (All)** to filter fields by computing logic status.
- Enter a keyword in the search box to search for fields.

Modify a field

 **Note** On the configuration tab of a logical dimension table, you can only modify fields that are associated with dimensions.

If you need to modify the primary key, move the pointer over the  icon in the **Actions** column and click **Details**. On the tab that appears, modify the parameters as needed. For more information, see [Modify a dimension](#).

To modify fields that are associated with dimensions, perform the following steps:

1. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
2. On the **Central Table Settings** tab, click **Group by Type** or **Group by Source** to view fields as needed.
3. On the **Group by Type** or **Group by Source** tab, find the field that you want to modify and click the  icon in the **Actions** column.
4. In the **Change Field** dialog box, modify the **Field Name**, **Field Display Name**, **Description**, and **Data Type** parameters as needed.
5. Click **OK**.
6. In the **Confirm Close** message, click **OK**.

7. Save, submit, and then publish the logical dimension table.
 - i. After you modify the field, click the  icon in the upper-right corner of the configuration tab to save the modified field.
 - ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
 - v. (Optional) Publish the logical dimension table.
 - If the current project is in Dev mode, publish the logical dimension table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical dimension table after you submit it.

Delete a field

 **Note** You can delete only the fields that are associated with dimensions.

1. On the configuration tab of the logical dimension table, click **Central Table Settings** in the top navigation bar.
2. On the **Central Table Settings** tab, click **Group by Type** or **Group by Source** to view fields as needed.
3. On the **Group by Type** or **Group by Source** tab, find the field that you want to delete and click the  icon in the **Actions** column.
4. In the message that appears, click **OK**.

9.9.1.4.3. Configure a logical table conversion task

To specify the retention period, partition fields, and custom parameters for a logical dimension table, you can configure a logical table conversion task for the table. This topic describes how to configure a logical table conversion task for a logical dimension table.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Dimension Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Dimension Tables section appears.

3. In the **Logical Dimension Tables** section, click the logical dimension table for which you want to configure a logical table conversion task. The configuration tab of the logical dimension table appears.
4. (Optional)Steal the lock of the logical dimension table.
 - o If the logical dimension table is locked by yourself, you do not need to steal the lock.
 - o If the logical dimension table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical dimension table, click **Logical Table Conversion Task Settings** in the top navigation bar.
6. In the **Logical Table Conversion Task Settings** pane, set the parameters as required.

Parameter	Description
Retention (Days)	The retention period of the logical dimension table. You can enter a value in the field or select one of the following options: 7, 14, 30, and 365 .
Select Partitioning Fields	The partition fields of the logical dimension table.
Custom Parameter Settings	The custom parameters to be specified for the logical dimension table.

7. Click **OK**.
8. Save, submit, and then publish the logical dimension table.
 - i. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to save the logical table conversion task.
 - ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
 - v. (Optional)Publish the logical dimension table.
 - If the current project is in Dev mode, publish the logical dimension table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical dimension table after you submit it.

9.9.1.4.4. Configure a scheduling policy

Dataphin allows you to configure the scheduling rules and dependencies of nodes to make sure that tasks can be properly scheduled. This topic describes how to configure a scheduling policy for a logical dimension table.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Dimension Tables** section.

- i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Dimension Tables section appears.
3. In the **Logical Dimension Tables** section, click the logical dimension table for which you want to configure a scheduling policy. The configuration tab of the logical dimension table appears.
 4. (Optional) Steal the lock of the logical dimension table.
 - o If the logical dimension table is locked by yourself, you do not need to steal the lock.
 - o If the logical dimension table is locked by another user, click the  icon in the upper-right corner to steal the lock.
 5. On the configuration tab of the logical dimension table, click **Scheduling Configuration** in the top navigation bar.
 6. In the **Scheduling Configuration** pane, set the parameters as required.
 - i. Set the parameters in the **Basic Information** section. Dataphin automatically generates values for the **Logical Table Name**, **Task Type**, and **Description** parameters. You cannot change these values.

Parameter	Description
Description	The description of the scheduling policy.
Priority	The priority based on which the logical table task is scheduled. Valid values: <ul style="list-style-type: none"> ▪ Lowest Priority ▪ Low Priority ▪ Medium Priority ▪ High Priority ▪ Highest Priority
Parameters	The specified values for the parameters that are used in the code of the task. You can click Parameters and Descriptions to view the rules for setting task parameters and the time parameters that are supported in the scheduling system of Dataphin.

- ii. Set the Depend on Previous Instance parameter in the **Recurrence** section.
 - If you set the **Depend on Previous Instance** parameter to **No**, the task of the logical dimension table does not depend on previous instances.
 - If you set the **Depend on Previous Instance** parameter to **Yes**, perform the following steps to configure dependencies:
 - a. Click the  icon.
 - b. Set the parameters as required.

No.	Description
1	Select the node type. You can select Select Task or Nodes of This Logical Table from the drop-down list.
2	<ul style="list-style-type: none"> ■ If you select Select Task, specify a node ID by using one of the following methods: <ul style="list-style-type: none"> ■ Click the  icon, enter a keyword to search for the desired node ID, and then select the node ID. ■ Click Enter a node ID and select a node ID from the drop-down list. ■ If you select Nodes of This Logical Table, click the  icon, select the fields to depend on, and then click OK.
3	<p>Select the fields to depend on by using one of the following methods:</p> <ul style="list-style-type: none"> ■ Click the  icon, enter a keyword to search for fields, select the desired fields, and then click OK. ■ Click Select Fields Depended On, select the desired fields, and then click OK.
4	To add more dependencies, click the  icon.

iii. Set the parameters in the **Dependency** section.

Parameter or section	Description
Automatic Parse	Specifies whether to automatically parse the dependencies of the logical table task. If you click Parse Input and Output , Dataphin automatically parses the task dependencies on instances of upstream nodes and the current node, and displays the dependency information in the Upstream Dependency and Logical Table Node (This Node) sections, respectively. If you click Parse Input and Output repeatedly, the system updates the information in the Upstream Dependency and Logical Table Node (This Node) sections.
Upstream Dependency	<p>The upstream nodes on which the physical node of the logical dimension table depends. You can add a dependent upstream node for the logical table task by using one of the following methods:</p> <ul style="list-style-type: none"> ■ Add a parsed node as a dependent upstream node. <ul style="list-style-type: none"> a. Click Parse Error Information. Source physical tables and other nodes related to the logical dimension table appear. b. Select a node and click Confirm Association. ■ Add a system node, for example, a zero-load node, as a dependent upstream node. <ul style="list-style-type: none"> a. Click Add Upstream Dependency. b. Select an upstream node and a source physical table. <p style="margin-left: 20px;">You can click the  icon to add more dependent upstream nodes.</p> c. Click OK.
Logical Table Node (This Node)	The output name of the logical table task. Dataphin automatically generates the output information about the task.

7. Click **OK**.

8. Save, submit, and then publish the logical dimension table.

- i. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to save the configured scheduling policy.
- ii. On the configuration tab of the logical dimension table, click the  icon in the upper-right corner to submit the logical dimension table.
- iii. In the dialog box that appears, enter your comments.
- iv. Click **OK**.
- v. (Optional) Publish the logical dimension table.
 - If the current project is in Dev mode, publish the logical dimension table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical dimension table after you submit it.

9.9.1.4.5. Unpublish and delete a logical dimension table

This topic describes how to unpublish and delete logical dimension tables in different states.

Prerequisites

Dimensions are created. For more information, see [Create a dimension](#).

Context

A logical dimension table may be in the following states:

- After you create and save a logical dimension table, it enters the **Draft** state.
- After you submit a logical dimension table, it enters the **Submitted** state.
- After you modify and save a logical dimension table in the **Submitted** state, it enters the **Developing** state.
- After you unpublish a logical dimension table in the **Submitted** state, it enters the **Draft** state.

Limits on unpublishing and deleting logical dimension tables:

- You can unpublish only the logical dimension tables in the **Developing** or **Submitted** state.
- You can delete only the logical dimension tables in the **Draft** state.

Unpublish a logical dimension table

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Dimension Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Dimension Tables section appears.
3. In the **Logical Dimension Tables** section, move the pointer over the  icon next to the logical dimension table that you want to unpublish and select **Unpublish**.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Delete a logical dimension table

1. In the **Logical Dimension Tables** section, move the pointer over the  icon next to the logical dimension table that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.1.5. Logical tables: Logical fact tables

9.9.1.5.1. Create a logical fact table

A logical fact table describes a business process in detail. This topic describes how to create a logical fact table.

Prerequisites

Data domains are created.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. Open the **Create Logical Fact Table** dialog box by using one of the following methods:
 - o In the left-side navigation pane, click the  icon next to the project name and choose **Logical Tables > Logical Fact Tables**.
 - o In the **Logical Fact Tables** section, click the  icon next to **Logical Fact Tables**.
 - o In the right-side workspace of the Develop page, click the  icon below **Logical Fact Table**.
4. In the **Create Logical Fact Table** dialog box, perform the following steps:

i. Set the parameters in the **Basic Information** step.

Parameter	Description
Data Domain	The data domain to which the business process to be described by the logical fact table belongs.
Business Process	The business process to be described by the logical fact table.
Fact Table Type	<p>The type of the logical fact table. Valid values: Transaction Fact Table and Periodic Snapshot Fact Table.</p> <ul style="list-style-type: none"> Transaction fact tables track and measure events that occur in business activities. For example, you can design a transaction fact table to track and measure the order payment event in order transactions. Periodic snapshot fact tables measure the status of entities at regular intervals. For example, you can use a periodic snapshot fact table to track the account balance or commodity inventory.
Name	<p>The name of the logical fact table. The value of the Name parameter is in the <code>fact_Business process name_Custom name_di</code> or <code>fact_Business process name_Custom name_df</code> format. The suffix of the name varies with the value of the Fact Table Type parameter.</p> <ul style="list-style-type: none"> If you set the Fact Table Type parameter to Transaction Fact Table, Dataphin automatically suffixes the name of the logical fact table with di. If you set the Fact Table Type parameter to Periodic Snapshot Fact Table, Dataphin automatically suffixes the name of the logical fact table with df.
Display Name	The display name of the logical fact table. The display name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the logical fact table.
Main Source Table	The main source table of the logical fact table. You can select a physical table or a logical table as the main source table.

ii. Click **Next**.

iii. Set the parameters in the **Primary Key Definition** step.

- If you set the **Set Primary Key** parameter to **No**, you can determine whether to set the **Filter Conditions** parameter based on your needs.
- If you set the **Set Primary Key** parameter to **Yes**, set the required parameters as described in the following table.

Parameter	Description
Field Name	The name of the primary key field. Example: item_id
Field Display Name	The display name of the primary key field. Example: commodity_id
Field Data Type	The data type of the primary key field.
Field Computing Logic	The computing logic of the primary key field.

 **Note** If you specify a unique primary key field, for example, order_id, for a logical fact table, you must associate other fields that are added to the logical fact table with the primary key field.

iv. Click **Submit**.

5. Add measures.

- i. On the configuration tab of the logical fact table, open the **Create Measure** dialog box by using one of the following methods:
 - Click **+ Add Measure** in the **Central Table** section.
 - Click **Central Table Settings** in the top navigation bar. On the **Central Table Settings** tab, move the pointer over **Create Field** and select **Measure**.
- ii. In the **Create Measure** dialog box, set the parameters as required.
 - If you set the **Source Table** parameter to **Import Fields**, perform the following steps to add fields: Click the  icon next to each desired field in the **Select New Fields** section to add the field to the **Create Field** section. Then, enter a display name for the field in the **Field Display Name** column. Click **Save and Verify**.
 - If you set the **Source Table** parameter to **Customize Fields**, enter SQL statements in the code editor to define fields. You can click **Example** in the upper part of the code editor to view the sample SQL statement. Click **Code Check** to verify the syntax of the SQL statements you entered. After the SQL statements pass the verification, click **Save and Verify**.

6. Add dimension associations.

- i. On the configuration tab of the logical fact table, open the **Create Dimension Association** dialog box by using one of the following methods:
 - Click **+ Create Dimension Association** in the **Central Table** section.
 - Click **Central Table Settings** in the top navigation bar. On the **Central Table Settings** tab, move the pointer over **Create Field** and select **Dimension Association**.

- ii. In the **Create Dimension Association** dialog box, set the parameters as required.

Parameter	Description
Associated Dimension	The dimension to be associated with the current logical fact table.
Edit Association Logic	The logic for associating a field in the current logical fact table with the specified dimension. Select a measure from the drop-down list under Fact Table Field to Be Associated .
Edit Dimension Role	<p>The role of the specified dimension.</p> <ul style="list-style-type: none"> ▪ Role Name: the role name of the dimension. By default, Dataphin sets this parameter to the name of the specified dimension. ▪ Role Display Name: the role display name of the dimension. By default, Dataphin sets this parameter to the display name of the specified dimension. <p>You can change the values of the Role Name and Role Display Name parameters based on your business needs.</p>
Default Value Settings	The default value that is used when the foreign key of the logical fact table cannot be associated with the specified dimension. By default, the Default Value parameter is set to -110.

7. Add fact attributes.

- i. On the configuration tab of the logical fact table, open the **Create Fact Attribute** dialog box by using one of the following methods:
- Click **+ Add Fact Attribute** in the **Central Table** section.
 - Click **Central Table Settings** in the top navigation bar. On the **Central Table Settings** tab, move the pointer over **Create Field** and select **Fact Attribute**.
- ii. In the **Create Fact Attribute** dialog box, set the parameters as required.
- If you set the **Source Table** parameter to **Import Fields**, perform the following steps to add fields: Click the  icon next to each desired field in the **Select New Fields** section to add the field to the **Create Field** section. Then, enter a display name for the field in the **Field Display Name** column. Click **Save and Verify**.
 - If you set the **Source Table** parameter to **Customize Fields**, enter SQL statements in the code editor to define fields. You can click **Example** in the upper part of the code editor to view the sample SQL statement. Click **Code Check** to verify the syntax of the SQL statements you entered. After the SQL statements pass the verification, click **Save and Verify**.

8. Submit and publish the logical fact table.

- i. On the configuration tab of the logical fact table, click the **Submit** icon in the upper-right corner to submit the logical fact table.
- ii. In the dialog box that appears, enter your comments.
- iii. Click **OK**.
- iv. (Optional) Publish the logical fact table.
- If the current project is in Dev mode, publish the logical fact table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical fact table after you submit it.

9.9.1.5.2. Configure a logical table conversion task

To specify the retention period, partition fields, and custom parameters for a logical fact table, you can configure a logical table conversion task for the table. This topic describes how to configure a logical table conversion task for a logical fact table.

Prerequisites

Logical fact tables are created. For more information, see [Create a logical fact table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. In the **Logical Fact Tables** section, move the pointer over the  icon next to the logical fact table for which you want to configure a logical table conversion task and select **Change**.
4. (Optional) Steal the lock of the logical fact table.
 - o If the logical fact table is locked by yourself, you do not need to steal the lock.
 - o If the logical fact table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical fact table, click **Logical Table Conversion Task Settings** in the top navigation bar.
6. In the **Logical Table Conversion Task Settings** pane, set the parameters as required.

Parameter	Description
Retention (Days)	The retention period of the logical fact table. You can enter a value in the field or select one of the following options: 7, 14, 30, and 365.
Select Partitioning Fields	The partition fields of the logical fact table.
Custom Parameter Settings	The custom parameters to be specified for the logical fact table.

7. Click **OK**.
8. Save, submit, and then publish the logical fact table.
 - i. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to save the logical table conversion task.

- ii. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to submit the logical fact table.
- iii. In the dialog box that appears, enter your comments.
- iv. Click **OK**.
- v. (Optional) Publish the logical fact table.
 - If the current project is in Dev mode, publish the logical fact table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical fact table after you submit it.

9.9.1.5.3. Modify a logical fact table

A logical fact table describes a business process in detail. This topic describes how to modify a logical fact table.

Prerequisites

Logical fact tables are created. For more information, see [Create a logical fact table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. In the **Logical Fact Tables** section, move the pointer over the  icon next to the logical fact table that you want to modify and select **Change**.
4. (Optional) Steal the lock of the logical fact table.
 - If the logical fact table is locked by yourself, you do not need to steal the lock.
 - If the logical fact table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical fact table, modify settings about the central table, logical table conversion task, and scheduling policy. For more information, see [Create a logical fact table](#), [Configure a scheduling policy](#), and [Configure a logical table conversion task](#).
6. Save, submit, and then publish the logical fact table.
 - i. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to save the logical fact table.
 - ii. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to submit the logical fact table.

- iii. In the dialog box that appears, enter your comments.
- iv. Click **OK**.
- v. (Optional) Publish the logical fact table.
 - If the current project is in Dev mode, publish the logical fact table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical fact table after you submit it.

9.9.1.5.4. Configure a scheduling policy

Dataphin allows you to configure the scheduling rules and dependencies of nodes to make sure that tasks can be properly scheduled. This topic describes how to configure a scheduling policy for a logical fact table.

Prerequisites

Logical fact tables are created. For more information, see [Create a logical fact table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. In the **Logical Fact Tables** section, click the logical fact table for which you want to configure a scheduling policy. The configuration tab of the logical fact table appears.
4. (Optional) Steal the lock of the logical fact table.
 - If the logical fact table is locked by yourself, you do not need to steal the lock.
 - If the logical fact table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical fact table, click **Scheduling Configuration** in the top navigation bar.
6. In the **Scheduling Configuration** pane, set the parameters as required.

- i. Set the parameters in the **Basic Information** section. Dataphin automatically generates values for the **Logical Table Name**, **Task Type**, and **Description** parameters. You cannot change these values.

Parameter	Description
Description	The description of the scheduling policy.
Priority	The priority based on which the logical table task is scheduled. Valid values: <ul style="list-style-type: none"> ■ Lowest Priority ■ Low Priority ■ Medium Priority ■ High Priority ■ Highest Priority
Parameters	The specified values for the parameters that are used in the code of the task. You can click Parameters and Descriptions to view the rules for setting task parameters and the time parameters that are supported in the scheduling system of Dataphin.

- ii. Set the **Depend on Previous Instance** parameter in the **Recurrence** section.
- If you set the **Depend on Previous Instance** parameter to **No**, the task of the logical fact table does not depend on previous instances.
 - If you set the **Depend on Previous Instance** parameter to **Yes**, perform the following steps to configure dependencies:
 - a. Click the  icon.

b. Set the parameters as required.

No.	Description
1	Select the node type. You can select Select Task or Nodes of This Logical Table from the drop-down list.
2	<ul style="list-style-type: none"> ■ If you select Select Task, specify a node ID by using one of the following methods: <ul style="list-style-type: none"> ■ Click the  icon, enter a keyword to search for the desired node ID, and then select the node ID. ■ Click Enter a node ID and select a node ID from the drop-down list. ■ If you select Nodes of This Logical Table, click the  icon, select the fields to depend on, and then click OK.
3	Select the fields to depend on by using one of the following methods: <ul style="list-style-type: none"> ■ Click the  icon, enter a keyword to search for fields, select the desired fields, and then click OK. ■ Click Select Fields Depended On, select the desired fields, and then click OK.
4	To add more dependencies, click the  icon.

iii. Set the parameters in the **Dependency** section.

Parameter or section	Description
Automatic Parse	Specifies whether to automatically parse the dependencies of the logical table task. If you click Parse Input and Output , Dataphin automatically parses the task dependencies on instances of upstream nodes and the current node, and displays the dependency information in the Upstream Dependency and Logical Table Node (This Node) sections, respectively. If you click Parse Input and Output repeatedly, the system updates the information in the Upstream Dependency and Logical Table Node (This Node) sections.
Upstream Dependency	<p>The upstream nodes on which the physical node of the logical fact table depends. You can add a dependent upstream node for the logical table task by using one of the following methods:</p> <ul style="list-style-type: none"> ■ Add a parsed node as a dependent upstream node. <ul style="list-style-type: none"> a. Click Parse Error Information. Source physical tables and other nodes related to the logical fact table appear. b. Select a node and click Confirm Association. ■ Add a system node, for example, a zero-load node, as a dependent upstream node. <ul style="list-style-type: none"> a. Click Add Upstream Dependency. b. Select an upstream node and a source physical table. <p style="margin-left: 20px;">You can click the  icon to add more dependent upstream nodes.</p> c. Click OK.
Logical Table Node (This Node)	The output name of the logical table task. Dataphin automatically generates the output information about the task.

7. Click **OK**.

8. Save, submit, and then publish the logical fact table.

- i. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to save the configured scheduling policy.
- ii. On the configuration tab of the logical fact table, click the  icon in the upper-right corner to submit the logical fact table.
- iii. In the dialog box that appears, enter your comments.
- iv. Click **OK**.
- v. (Optional) Publish the logical fact table.
 - If the current project is in Dev mode, publish the logical fact table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical fact table after you submit it.

9.9.1.5.5. View historical version information

This topic describes how to view historical version information about a logical fact table.

Prerequisites

Logical fact tables are created. For more information, see [Create a logical fact table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. In the **Logical Fact Tables** section, move the pointer over the  icon next to the logical fact table for which you want to view historical version information and select **Change**.
4. On the configuration tab of the logical fact table, click **Versions** in the top navigation bar.
5. In the **Versions** pane, view the historical version information about the logical fact table.
6. Find a version and click **Details** in the **Actions** column.
7. On the page that appears, you can view the basic information, scheduling information, and code details about the version. You can also view the settings of the logical table conversion task in this version.

9.9.1.5.6. Unpublish and delete a logical fact table

This topic describes how to unpublish and delete logical fact tables in different states.

Prerequisites

Logical fact tables are created. For more information, see [Create a logical fact table](#).

Context

A logical fact table may be in the following states:

- After you create and save a logical fact table, it enters the **Draft** state.
- After you submit a logical fact table, it enters the **Submitted** state.
- After you modify and save a logical fact table in the **Submitted** state, it enters the **Developing** state.
- After you unpublish a logical fact table in the **Submitted** state, it enters the **Draft** state.

Limits on unpublishing and deleting logical fact tables:

- You can unpublish only the logical fact tables in the **Developing** or **Submitted** state.
- You can delete only the logical fact tables in the **Draft** state.

Unpublish a logical fact table

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Fact Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Fact Tables section appears.
3. In the **Logical Fact Tables** section, move the pointer over the  icon next to the logical fact table that you want to unpublish and select **Unpublish**.
 4. In the **Tip** dialog box, enter your comments.
 5. Click **OK**.

Delete a logical fact table

1. In the **Logical Fact Tables** section, move the pointer over the  icon next to the logical fact table that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.1.6. Data standardization: Atomic metrics

9.9.1.6.1. Create an atomic metric

An atomic metric is an abstraction of statistical criteria and specific algorithms. Dataphin introduces the concept of development automation. To define a metric, you must specify its statistical criteria, which is the computing logic. This increases development efficiency and ensures the consistency of statistical results.

Context

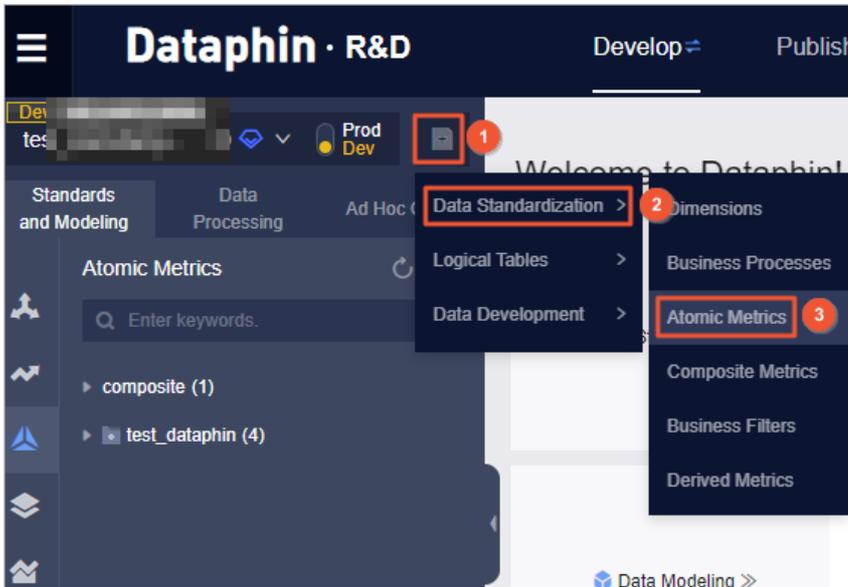
Dataphin allows you to create atomic metrics and composite metrics based on different computing logic. Before you create an atomic metric or a composite metric, take note of the following descriptions:

- An atomic metric is a metric whose value is directly obtained, for example, a payment amount.
- A composite metric is a metric whose value is calculated based on submitted atomic metrics and specified computing logic. For example, you have submitted two atomic metrics. One atomic metric measures the total payment amount and the other measures the number of customers who made the payments. Then, you can create a composite metric that uses the two atomic metrics to calculate the average payment amount per customer.

Create an atomic metric

1. [Log on to the Dataphin console.](#)
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.

- iv. On the Standards and Modeling tab, move the pointer over the left-side navigation submenu and click the  icon. The Atomic Metrics section appears.
3. Go to the **Create Atomic Metric** tab by using one of the following methods:
- o In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Atomic Metrics**.



- o In the Atomic Metrics section, click the  icon next to **Atomic Metrics** and select **Create Atomic Metric**.
 - o In the right-side workspace of the Develop page, click the  icon below **Atomic Metric** and select **Create Atomic Metric**.
 - o In the Atomic Metrics section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears. Click **Create Atomic Metric** in the upper-right corner and select **Atomic Metrics**.
4. On the **Create Atomic Metric** tab, select a data domain and a source table in the **Select Source Information** section.

Parameter	Description
Data Domain	The data domain of the business process to which the source table that you want to select belongs.
Source Table	<p>The source table that contains the fields that you want to reference in the computing logic of the atomic metric. To ensure that all atomic metric models are standard, Dataphin supports only logical dimension tables and logical fact tables as the source table.</p> <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p> Note After you select the source table, a list of existing atomic metrics that are defined based on the source table appears in the Atomic Metrics section.</p> </div>

- 5. In the **Atomic Metrics** section, click **Create Atomic Metric**.
- 6. In the **Create Atomic Metric** dialog box, set the parameters as required.

Create Atomic Metric
✕

* Primary Source Field ?

* Name * Display Name

Description 0/128

Data Type

Statistical Period Indicator ?

* Computing Logic ? Accumulable: Yes No

Beautify
Example
Code Check
?

```
1 fct_order_pay_df.city='shanghai'
```

Parameter	Description
Primary Source Field	The primary source field of the atomic metric. You can select Entire Table or a field in the source table from the drop-down list.
Name	The name of the atomic metric. The name of the selected primary source field is automatically entered in the Name field. You can modify the name as needed.
Display Name	The display name of the atomic metric. The display name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the atomic metric.
Data Type	The data type of the atomic metric. Valid values: STRING , BIGINT , DOUBLE , DATETIME , and DECIMAL .
Field	The field to be associated with the statistical period. By default, the time-based partition field in the business unit to which the atomic metric belongs is selected.
Format Type	The format of the value of the field associated with the statistical period when the value is used in computation.
Computing Logic	<p>The computing logic of the atomic metric. Enter an SQL statement, for example, count(distinct order_id), to define the atomic metric based on the source logical table. In the SQL statement, order_id is a field in the source table.</p> <p>Perform the following steps to define the computing logic:</p> <ol style="list-style-type: none"> i. Click Example to view the description about the computing logic of atomic metrics and the sample SQL statement. ii. After you enter an SQL statement, click Code Check to verify the SQL statement you entered.

> Document Version: 20210621

2689

7. Submit and publish the atomic metric.
 - i. After you set the parameters, click **Submit**.
 - ii. (Optional) Publish the atomic metric.
 - If the current project is in Dev mode, publish the atomic metric to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the atomic metric. The atomic metric can be used for scheduling after you submit it.

Create a composite metric

1. Open the **Create Composite Metric** dialog box by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Composite Metrics**.
 - In the **Atomic Metrics** section, click the  icon next to Atomic Metrics and select **Create Composite Metric**.
 - In the right-side workspace of the Develop page, click the  icon below **Atomic Metric** and select **Create Composite Metric**.
 - In the Atomic Metrics section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears. Click **Create Atomic Metric** in the upper-right corner and select **Composite**.
2. In the **Create Composite Metric** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the composite metric. The name can contain letters, digits, and underscores (_).
Display Name	The display name of the composite metric. The display name can contain letters, digits, underscores (_), and hyphens (-).

Parameter	Description
Description	The description of the composite metric.
Data Type	The data type of the composite metric. Valid values: STRING , BIGINT , DOUBLE , DATETIME , and DECIMAL .
Computing Logic	<p>The computing logic of the composite metric. When you define the computing logic, you can use submitted atomic metrics as needed.</p> <ul style="list-style-type: none"> ○ In the code editor, enter an SQL statement in the form of a formula that uses submitted atomic metrics to define the computing logic of the composite metric. ○ <ul style="list-style-type: none"> a. Click the  icon next to Referable Atomic Metrics and select a data domain and a source table. b. In the atomic metric list that appears, click Add next to an atomic metric to reference the atomic metric in the SQL statement you entered. <p>You can also enter a keyword in the search box next to the  icon to search for atomic metrics. In the atomic metric list that appears, click Add next to an atomic metric to reference the atomic metric in the SQL statement you entered.</p> <p>For example, <code>crt_amt</code> and <code>usr_cnt</code> are two submitted atomic metrics that measure the total payment amount and the number of customers who made the payments, respectively. You can enter <code>crt_amt/usr_cnt</code> in the code editor to define a composite metric that calculates the average payment amount per customer.</p>

3. After you set the parameters, click **Submit**.
4. (Optional) Publish the composite metric.
 - If the current project is in Dev mode, publish the composite metric to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the composite metric. The composite metric can be used for scheduling after you submit it.

9.9.1.6.2. Modify an atomic metric

An atomic metric is an abstraction of statistical criteria and specific algorithms. To define an atomic metric, you must specify its statistical criteria, which is the computing logic. This topic describes how to modify an atomic metric.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Context

After you modify the computing logic of an atomic metric, the modification applies to all the derived metrics that are created based on the atomic metric. Before you modify an atomic metric, you must consider the impact on the related derived metrics.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Atomic Metrics** section.

- i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Atomic Metrics section appears.
3. Open the **Change Atomic Metric** dialog box by using one of the following methods:
 - o In the Atomic Metrics section, move the pointer over the  icon next to the atomic metric that you want to modify and select **Change**.
 - o In the Atomic Metrics section, click the atomic metric that you want to modify. On the **View Attributes** tab, click **Change**.
 - a. In the Atomic Metrics section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears.
 - b. Find the atomic metric that you want to modify and click the  icon in the **Actions** column.
 4. In the **Change Atomic Metric** dialog box, modify the parameters as needed. For more information, see [Create an atomic metric](#).
 5. After you modify the parameters, click **Submit**.
 6. (Optional) Publish the atomic metric.
 - o If the current project is in Dev mode, publish the atomic metric to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the atomic metric. The atomic metric can be used for scheduling after you submit it.

9.9.1.6.3. View atomic metrics that share the same source table

Dataphin allows you to create multiple atomic metrics based on the same source logical table. This topic describes how to view atomic metrics that share the same source table.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Context

Dataphin allows you to view the atomic metrics that share the same source table with a specific atomic metric.

 **Note** Composite metrics do not share the same source table with each other.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Atomic Metrics** section appears.
3. In the **Atomic Metrics** section, click the atomic metric that want to view. On the **View Attributes** tab, click **View Same Source**.
 4. On the **Same-Source Atomic Metrics** tab, view the atomic metrics that share the same source table with the atomic metric.

9.9.1.6.4. Clone an atomic metric

Dataphin allows you to clone an atomic metric. This operation creates an atomic metric that shares the same source table with an existing atomic metric. This topic describes how to clone an atomic metric.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Atomic Metrics** section appears.
3. In the **Atomic Metrics** section, click the atomic metric that you want to clone. On the **View Attributes** tab, move the pointer over the  icon and select **Clone**.
4. In the **Clone Atomic Metric** dialog box, set the parameters as required. For more information, see [Create an atomic metric](#).
5. Click **Submit**.
6. In the dialog box that appears, enter your comments.
7. Click **OK**.
8. (Optional) Publish the atomic metric.
 - o If the current project is in **Dev** mode, publish the atomic metric to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - o If the current project is in **Basic** mode, you do not need to publish the atomic metric. The atomic metric can be used for scheduling after you submit it.

9.9.1.6.5. View derived metrics created based on an atomic metric

Dataphin allows you to view derived metrics that are created based on an atomic metric. This topic describes how to view derived metrics of an atomic metric.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Context

Dataphin allows you to create derived metrics based on atomic metrics in the **Submitted** and **Developing** states. You can view derived metrics of only the atomic metrics in the **Submitted** and **Developing** states.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Atomic Metrics** section appears.
3. In the **Atomic Metrics** section, click the atomic metric for which you want to view derived metrics. On the **View Attributes** tab, move the pointer over the  icon and select **Related Derived Metrics**.
4. On the **Derived Metrics** tab of the **Object Explorer** tab, view all the derived metrics that are created based on the atomic metric.

9.9.1.6.6. Create a derived metric

Dataphin allows you to create a derived metric based on an atomic metric. This topic describes how to create a derived metric based on an atomic metric.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.

- iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Atomic Metrics** section appears.
3. In the **Atomic Metrics** section, click the atomic metric based on which you want to create a derived metric. On the **View Attributes** tab, move the pointer over the  icon and select **Create Derived Metric**.
 4. On the **Create Derived Metric** tab, set the parameters as required. For more information, see [Create a derived metric](#).
 5. Submit the derived metric.
 - i. After you set the parameters, click **Preview Derived Metric**.
 - ii. Click **Submit**.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
 6. (Optional) Publish the derived metric.
 - o If the current project is in Dev mode, publish the derived metric to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the derived metric. The derived metric can be used for scheduling after you submit it.

9.9.1.6.7. Unpublish and delete an atomic metric

This topic describes how to unpublish, unpublish and delete, and delete atomic metrics in different states.

Prerequisites

Atomic metrics are created. For more information, see [Create an atomic metric](#).

Context

- An atomic metric may be in the following states:
 - o After you create and save an atomic metric, it enters the **Draft** state.
 - o After you submit an atomic metric, it enters the **Submitted** state.
 - o After you modify and save an atomic metric in the **Submitted** state, it enters the **Developing** state.
 - o After you unpublish an atomic metric in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing atomic metrics:
 - o You can unpublish only the atomic metrics in the **Developing** or **Submitted** state.
 - o Only the super administrator, project administrator, and developer can unpublish atomic metrics.
 - o Before you unpublish an atomic metric, make sure that the atomic metric is not used by any derived metrics or other atomic metrics.
- Limits on deleting atomic metrics:
 - o You can delete only the atomic metrics in the **Draft** state.
 - o Only the super administrator, project administrator, and developer can delete atomic metrics.
- Limits on unpublishing and deleting atomic metrics:
 - o You can unpublish and delete only the atomic metrics in the **Developing** or **Submitted** state.
 - o Only the super administrator, project administrator, and developer can unpublish and delete atomic metrics.

- Before you unpublish and delete an atomic metric, make sure that the atomic metric is not used by any derived metrics or other atomic metrics.

Delete an atomic metric

1. [Log on to the Dataphin console.](#)
2. Go to the **Atomic Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Atomic Metrics** section appears.
3. Open the **Tip** dialog box for deleting an atomic metric by using one of the following methods:
 - In the **Atomic Metrics** section, move the pointer over the  icon next to the atomic metric that you want to delete and select **Delete**.
 - In the **Atomic Metrics** section, click the atomic metric that you want to delete. On the **View Attributes** tab, move the pointer over the  icon and select **Delete**.
 - a. In the **Atomic Metrics** section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears.
 - b. Find the atomic metric that you want to delete and choose  > **Delete** in the **Actions** column.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Unpublish an atomic metric

1. Open the **Tip** dialog box for unpublishing an atomic metric by using one of the following methods:
 - In the **Atomic Metrics** section, move the pointer over the  icon next to the atomic metric that you want to unpublish and select **Unpublish**.
 - In the **Atomic Metrics** section, click the atomic metric that you want to unpublish. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish**.
 - a. In the **Atomic Metrics** section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears.
 - b. Find the atomic metric that you want to unpublish and choose  > **Unpublish** in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

Unpublish and delete an atomic metric

1. Open the **Tip** dialog box for unpublishing and deleting an atomic metric by using one of the following methods:
 - In the **Atomic Metrics** section, move the pointer over the  icon next to the atomic metric that you want

- to unpublish and delete and select **Unpublish and Delete**.
- In the Atomic Metrics section, click the atomic metric that you want to unpublish and delete. On the View Attributes tab, move the pointer over the  icon and select **Unpublish and Delete**.
- a. In the Atomic Metrics section, click **Atomic Metrics Object List** in the lower part. The **Atomic Metrics** tab of the **Object Explorer** tab appears.
 - b. Find the atomic metric that you want to unpublish and delete and choose  > **Unpublish and Delete** in the **Actions** column.
- 2. In the **Tip** dialog box, enter your comments.
- 3. Click **OK**.

9.9.1.7. Data standardization: Business filters

9.9.1.7.1. Create a business filter

A business filter is used to define the scope of business to be measured. This topic describes how to create a business filter.

Prerequisites

Data domains are created. For more information, see [Create a data domain](#).

Context

- Dataphin allows you to create business filters based on logical dimension tables or logical fact tables.
- Business filters are used to define the conditions for derived metrics in a standardized manner, whereas atomic metrics are used to define the computing logic of derived metrics in a standardized manner.
- To make sure that derived metrics can be created in a uniform and standard manner, you must create business filters for a business unit based on the same logical table.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Business Filters** section appears.
3. Go to the **Create Business Filter** tab by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Business Filters**.
 - In the **Business Filters** section, click the  icon next to **Business Filters**.
 - In the right-side workspace of the **Develop** page, click the  icon below **Business Filter**.

- In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Click **Create Business Filter**.
- 4. On the **Create Business Filter** tab, set the **Data Domain** and **Source Table** parameters in the **Select Source Information** section.
- 5. In the **Business Filter Source** section, click **Create Business Filter**.
- 6. In the **Create Business Filter** dialog box, set the parameters as required.

Parameter	Description
Primary Source Field	The primary source field of the business filter.
Name	The name of the business filter. The name can contain letters, digits, and underscores (_).
Display Name	The display name of the business filter. The display name can contain letters, digits, underscores (_), and hyphens (-).
Description	The description of the business filter.
Computing Logic	<p>The computing logic of the business filter.</p> <ul style="list-style-type: none"> i. Click Example to view the description about the computing logic of business filters and the sample SQL statement. <div style="background-color: #f0f0f0; padding: 5px; margin: 5px 0;"> <pre>select province from dataphin_test where ds='\${bizdate}';</pre> </div> <ul style="list-style-type: none"> ii. Click Code Check to verify the SQL statements you entered.

- 7. Submit the business filter.
 - i. After you set the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
- 8. (Optional) Publish the business filter.
 - If the current project is in Dev mode, publish the business filter to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the business filter after you submit it.

9.9.1.7.2. Clone a business filter

This topic describes how to clone a business filter.

Prerequisites

Business filters are created. For more information, see [Create a business filter](#).

Context

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Filters section appears.
3. Open the **Clone Business Filter** dialog box by using one of the following methods:
 - o In the **Business Filters** section, click the data domain to which the business filter that you want to clone belongs. Click the business filter. On the **View Attributes** tab, move the pointer over the  icon and select **Clone**.
 - o In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter that you want to clone and choose  > **Clone** in the **Actions** column.
4. In the **Clone Business Filter** dialog box, set the parameters as required.
5. Submit the business filter.
 - i. After you set the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the business filter.
 - o If the current project is in **Dev** mode, publish the business filter to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - o If the current project is in **Basic** mode, you do not need to publish the business filter after you submit it.

9.9.1.7.3. View business filters that share the same source table

Dataphin allows you to create multiple business filters based on the same source logical table. This topic describes how to view business filters that share the same source table.

Prerequisites

Business filters are created. For more information, see [Create a business filter](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Filters section appears.

3. Go to the **Same-Source Business Filters** tab by using one of the following methods:
 - In the **Business Filters** section, click the data domain to which the business filter for which you want to view business filters that share the same source table belongs. Click the business filter. On the View Attributes tab, click **View Same Source**.
 - In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter for which you want to view business filters that share the same source table and click the  icon in the **Actions** column.
4. On the **Same-Source Business Filters** tab, view the business filters that share the same source table with the business filter.

9.9.1.7.4. View derived metrics created based on a business filter

Dataphin allows you to view derived metrics that are created based on a business filter. This topic describes how to view derived metrics of a business filter.

Prerequisites

Business filters are created. For more information, see [Create a business filter](#).

Context

Dataphin allows you to create derived metrics based on business filters in the **Submitted** and **Developing** states. You can view derived metrics of only the business filters in the **Submitted** and **Developing** states.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Business Filters** section appears.
3. Go to the **Derived Metrics** tab by using one of the following methods:
 - In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter for which you want to view derived metrics and click the  icon in the **Actions** column.
 - In the **Business Filters** section, click the data domain to which the business filter for which you want to view derived metrics belongs. Click the business filter. On the View Attributes tab, move the pointer over the  icon and select **Related Derived Metrics**.
4. On the **Derived Metrics** tab of the **Object Explorer** tab, view all the derived metrics that are created based on the business filter.

9.9.1.7.5. Modify a business filter

A business filter is used to define the scope of business to be measured. This topic describes how to modify a business filter.

Prerequisites

Business filters are created. For more information, see [Create a business filter](#).

Context

After the computing logic of a business filter is modified, the statistical analysis logic for all the derived metrics that are created based on the business filter is also updated.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Business Filters** section appears.
3. Open the **Change Business Filter** dialog box by using one of the following methods:
 - o In the **Business Filters** section, click the data domain to which the business filter that you want to modify belongs. Click the business filter. On the **View Attributes** tab, click **Change**.
 - o In the **Business Filters** section, click the data domain to which the business filter that you want to modify belongs. Move the pointer over the  icon next to the business filter and select **Change**.
 - o In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter that you want to modify and click the  icon in the **Actions** column.
4. In the **Change Business Filter** dialog box, modify the parameters as needed. For more information, see [Create a business filter](#).
5. Submit the business filter.
 - i. After you modify the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the business filter.
 - o If the current project is in **Dev** mode, publish the business filter to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - o If the current project is in **Basic** mode, you do not need to publish the business filter after you submit it.

9.9.1.7.6. Unpublish and delete a business filter

This topic describes how to unpublish, unpublish and delete, and delete business filters in different states.

Prerequisites

Business filters are created. For more information, see [Create a business filter](#).

Context

- A business filter may be in the following states:
 - After you create and save a business filter, it enters the **Draft** state.
 - After you submit a business filter, it enters the **Submitted** state.
 - After you modify and save a business filter in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a business filter in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing business filters:
 - You can unpublish only the business filters in the **Developing** or **Submitted** state.
 - You can unpublish only the business filters without derived metrics.
- Limits on deleting business filters: You can delete only the business filters in the **Draft** state.
- Limits on unpublishing and deleting business filters:
 - You can unpublish and delete only the business filters in the **Developing** or **Submitted** state.
 - You can unpublish and delete only the business filters without derived metrics.

Delete a business filter

1. [Log on to the Dataphin console](#).
2. Go to the **Business Filters** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project. You can skip this step if the current project is in **Dev** or **Basic** mode and is not the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Business Filters section appears.
3. Open the **Tip** dialog box for deleting a business filter by using one of the following methods:
 - In the **Business Filters** section, click the data domain to which the business filter that you want to delete belongs. Move the pointer over the  icon next to the business filter and select **Delete**.
 - In the **Business Filters** section, click the data domain to which the business filter that you want to delete belongs. Click the business filter. On the **View Attributes** tab, move the pointer over the  icon and select **Delete**.
 - In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter that you want to delete and choose  > **Delete** in the **Actions** column.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Unpublish a business filter

1. Open the **Tip** dialog box for unpublishing a business filter by using one of the following methods:
 - In the **Business Filters** section, click the data domain to which the business filter that you want to unpublish belongs. Move the pointer over the  icon next to the business filter and select **Unpublish**.

- In the **Business Filters** section, click the data domain to which the business filter that you want to unpublish belongs. Click the business filter. On the View Attributes tab, move the pointer over the  icon and select **Unpublish**.
 - In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter that you want to unpublish and choose  > **Unpublish** in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
 3. Click **OK**.

Unpublish and delete a business filter

1. Open the **Tip** dialog box for unpublishing and deleting a business filter by using one of the following methods:
 - In the **Business Filters** section, click the data domain to which the business filter that you want to unpublish and delete belongs. Move the pointer over the  icon next to the business filter and select **Unpublish and Delete**.
 - In the **Business Filters** section, click **Business Filters Object List** in the lower part. The **Business Filters** tab of the **Object Explorer** tab appears. Find the business filter that you want to unpublish and delete and choose  > **Unpublish and Delete** in the **Actions** column.
 - In the **Business Filters** section, click the data domain to which the business filter that you want to unpublish and delete belongs. Click the business filter. On the View Attributes tab, move the pointer over the  icon and select **Unpublish and Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.1.8. Data standardization: Derived metrics

9.9.1.8.1. Create a derived metric

Derived metrics are used to define the scope of business for atomic metrics. This topic describes how to create a derived metric.

Prerequisites

Data domains are created. For more information, see [Create a data domain](#).

Context

Dataphin allows you to create derived metrics in a standard and unambiguous manner. Note the following information about derived metrics:

- A derived metric consists of the following elements: atomic metric, business filter, statistical period, and statistic granularity.
- Dataphin allows you to create derived metrics based on logical dimension tables or logical fact tables.
- A derived metric belongs to only one source logical table in a business unit.
- An atomic metric and a derived metric may belong to logical tables in different data domains. Therefore, a derived metric may belong to multiple data domains.

Procedure

1. Log on to the Dataphin console.
2. Go to the **Derived Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Derived Metrics** section appears.
3. Go to the **Create Derived Metric** tab by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Standardization > Derived Metrics**.
 - In the **Derived Metrics** section, click the  icon next to **Derived Metrics**.
 - In the right-side workspace of the **Develop** page, click the  icon below **Derived Metric**.
 - In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Click **Create Derived Metric**.
4. Create a derived metric.
 - i. On the **Create Derived Metric** tab, select an atomic metric.
 - ii. Click **Next**.
 - iii. In the **Define Derived Metric** step, set the parameters as required.

Parameter	Description
Granularity	The statistic granularity of the derived metric. You can click Add Statistic Granularity to add statistic granularities for the derived metric. A derived metric supports at most three statistic granularities.
Statistical Period	The statistical period of the derived metric. You can click Add Statistical Period to add statistical periods for the derived metric. A derived metric supports at most three statistical periods.
Business Filter	The business filter of the derived metric. You can click Add Business Filter to add business filters for the derived metric. A derived metric supports at most three business filters.

- iv. After you set the parameters, click **Preview Derived Metric**.
5. (Optional) In the **Change Derived Metric** section, set the **Derived Metric Name** and **Derived Metric Display Name** parameters.
6. Save and submit the derived metric.
 - i. Click **Save** to save the derived metric.
 - ii. Click **Submit** to submit the derived metric.
 - iii. In the dialog box that appears, enter your comments.
 - iv. Click **OK**.
7. (Optional) Publish the derived metric.
 - If the current project is in **Dev** mode, publish the derived metric to the corresponding project in **Prod**

mode. For more information, see [Publishing management](#).

- If the current project is in Basic mode, you do not need to publish the derived metric after you submit it.

9.9.1.8.2. Modify a derived metric

Derived metrics are used to define the scope of business for atomic metrics. This topic describes how to modify a derived metric.

Prerequisites

Derived metrics are created. For more information, see [Create a derived metric](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Derived Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Derived Metrics** section appears.
3. Open the **Change Derived Metric** dialog box by using one of the following methods:
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to modify belongs. Click the derived metric. On the **View Attributes** tab, click **Change**.
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to modify belongs. Move the pointer over the  icon next to the derived metric that you want to modify and select **Change**.
 - In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Find the derived metric that you want to modify and click the  icon in the **Actions** column.
4. In the **Change Derived Metric** dialog box, modify the parameters as needed. For more information, see [Create a business filter](#).
5. (Optional) Save the derived metric. If you do not want to submit the modified derived metric, click **Save**.
6. Submit the derived metric.
 - i. Click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
7. (Optional) Publish the derived metric.
 - If the current project is in **Dev** mode, publish the derived metric to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the derived metric after you submit it.

9.9.1.8.3. Unpublish and delete a derived metric

This topic describes how to unpublish, unpublish and delete, and delete derived metrics in different states.

Prerequisites

Derived metrics are created. For more information, see [Create a derived metric](#).

Context

- A derived metric may be in the following states:
 - After you create and save a derived metric, it enters the **Draft** state.
 - After you submit a derived metric, it enters the **Submitted** state.
 - After you modify and save a derived metric in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a derived metric in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing, deleting, and unpublishing and deleting derived metrics:
 - You can unpublish only the derived metrics in the **Developing** or **Submitted** state.
 - You can delete only the derived metrics in the **Draft** state.
 - You can unpublish and delete only the derived metrics in the **Developing** or **Submitted** state.

Delete a derived metric

1. [Log on to the Dataphin console](#).
2. Go to the **Derived Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Derived Metrics** section appears.
3. Open the **Tip** dialog box for deleting a derived metric by using one of the following methods:
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to delete belongs. Move the pointer over the  icon next to the derived metric that you want to delete and select **Delete**.
 - In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Find the derived metric that you want to delete and click the  icon in the **Actions** column.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Unpublish a derived metric

1. Open the **Tip** dialog box for unpublishing a derived metric by using one of the following methods:
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to unpublish belongs. Move the pointer over the  icon next to the derived metric that you want to unpublish and select **Unpublish**.
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to unpublish belongs. Click the derived metric. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish**.

- In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Find the derived metric that you want to unpublish and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
 3. Click **OK**.

Unpublish and delete a derived metric

1. Open the **Tip** dialog box for unpublishing and deleting a derived metric by using one of the following methods:
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to unpublish and delete belongs. Move the pointer over the  icon next to the derived metric that you want to unpublish and delete and select **Unpublish and Delete**.
 - In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Find the derived metric that you want to unpublish and delete and click the  icon in the **Actions** column.
 - In the **Derived Metrics** section, click the data domain to which the derived metric that you want to unpublish and delete belongs. Click the derived metric. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish and Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.1.8.4. View the logical aggregate table associated with a derived metric

Dataphin allows you to view the logical aggregate table that is associated with a derived metric. This topic describes how to view the logical aggregate table that is associated with a derived metric.

Prerequisites

Derived metrics are created. For more information, see [Create a derived metric](#).

Context

Dataphin can aggregate only the derived metrics in the **Submitted** state to logical aggregate tables. Therefore, you can view only the logical aggregate table that is associated with a derived metric in the **Submitted** state.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Derived Metrics** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.

- iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Derived Metrics section appears.
3. Go to the details page of the logical aggregate table that is associated with a derived metric by using one of the following methods:
 - o In the **Derived Metrics** section, click **Derived Metrics Object List** in the lower part. The **Derived Metrics** tab of the **Object Explorer** tab appears. Find the derived metric for which you want to view the associated logical aggregate table and choose  > **View Aggregate Table** in the **Actions** column.
 - o In the **Derived Metrics** section, click the data domain to which the derived metric for which you want to view the associated logical aggregate table belongs. Click the derived metric. On the **View Attributes** tab, click **View Relative Aggregate Tables**.
4. On the tab that appears, view information about the logical aggregate table that is associated with the derived metric.

9.9.1.9. Logical tables: Logical aggregate tables

9.9.1.9.1. Create a logical aggregate table

A logical aggregate table stores statistics about a statistic granularity. This topic describes how to create a logical aggregate table.

Prerequisites

- Dimensions are created. For more information, see [Create a dimension](#).
- Derived metrics are created. For more information, see [Create a derived metric](#).

Context

Note the following information about logical aggregate tables:

- A logical aggregate table consists of statistical metrics at a specific statistic granularity.
- All statistical metrics in a logical aggregate table have the same statistic granularity.
- If the statistic granularity of a derived metric is the same as that of a logical aggregate table, Dataphin automatically associates the derived metric with this logical aggregate table.

The statistical metrics of a logical aggregate table are categorized into the following types:

- Derived metrics that are automatically aggregated by Dataphin
- Custom metrics

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Aggregate Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Aggregate Tables section appears.

3. Open the **Create Logical Aggregate Table** dialog box by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and choose **Logical Tables > Logical Aggregate Tables**.
 - In the **Logical Aggregate Tables** section, click the  icon next to **Logical Aggregate Tables**.
 - In the right-side workspace of the Develop page, click the  icon below **Logical Aggregate Table**.
4. In the **Create Logical Aggregate Table** dialog box, perform the following steps:
 - i. In the **Primary Key Information** step, set the **Granularity** parameter. You can select one or more submitted or published dimensions as the statistic granularity.
 - ii. Click **Next**.
 - iii. In the **Logical Aggregate Table Description** step, enter the description of the logical aggregate table.
5. Submit the logical aggregate table.
 - i. After you set the parameters in the **Create Logical Aggregate Table** dialog box, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the logical aggregate table.
 - If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

9.9.1.9.2. Modify a logical aggregate table

A logical aggregate table stores statistics about a statistic granularity. This topic describes how to modify a logical aggregate table.

Prerequisites

Logical aggregate tables are created. For more information, see [Create a logical aggregate table](#).

Context

You can perform the following operations on a logical aggregate table:

- Add derived metrics.
- Add custom metrics.
- Modify derived metrics.
- View details of derived metrics and custom metrics.
- Move derived metrics or custom metrics to a specified category.
- Delete custom metrics.

Add a derived metric

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Aggregate Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Aggregate Tables section appears.
3. In the **Logical Aggregate Tables** section, move the pointer over the  icon next to the logical aggregate table that you want to modify and select **Change**.
4. Steal the lock of the logical aggregate table.
 - o If the logical aggregate table is locked by yourself, you do not need to steal the lock.
 - o If the logical aggregate table is locked by another user, click the  icon in the upper-right corner to steal the lock.
5. On the configuration tab of the logical aggregate table, click the **Derived Metrics** tab.
6. On the **Derived Metrics** tab, click **Create Derived Metric**.
7. Create, submit, and publish a derived metric. For more information, see [Create a logical aggregate table](#).
8. Submit and publish the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
 - iv. Publish the logical aggregate table.
 - If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

Modify a derived metric

1. On the **Derived Metrics** tab of the configuration tab of the logical aggregate table, find the derived metric that you want to modify and click **Details** in the **Actions** column. On the **View Attributes** tab, click **Change**.
2. In the **Change Derived Metric** dialog box, set the parameters and then submit and publish the derived metric. For more information, see [Modify a derived metric](#).
3. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
4. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

View details of a derived metric

1. On the **Derived Metrics** tab of the configuration tab of the logical aggregate table, find the derived metric for which you want to view details and click **Details** in the **Actions** column.
2. On the View Attributes tab, view information about the derived metric, including the business unit and project to which it belongs and its associated atomic metrics.

Move a derived metric to a specified category

1. On the **Derived Metrics** tab of the configuration tab of the logical aggregate table, find the derived metric that you want to move and click **Category** in the **Actions** column.
2. In the **Categorize Metric** dialog box, enter a group name.
3. Click **Create** to create a group.
4. Set the **Categorize To** parameter to the created group and click **OK**.
5. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

Add a custom metric

1. On the configuration tab of the logical aggregate table, click the **Custom Metrics** tab.
2. On the **Custom Metrics** tab, click **Create Custom Metric**.
3. In the **Create Custom Metric** dialog box, perform the following steps:
 - i. In the **Primary Key Information** step, select the source physical table and set the primary key and association logic.
 - ii. Click **Associate and Go to Next Step**.
 - iii. In the **Logical Aggregate Table Description** step, select a field from the section on the left and click the > sign.
 - iv. Click **OK**.
4. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
5. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

View details of a custom metric

1. On the **Custom Metrics** tab of the configuration tab of the logical aggregate table, find the custom metric for which you want to view details and click **Details** in the **Actions** column.
2. On the View Attributes tab, view information on the **Table Structure** and **Table Information** tabs.

Delete a custom metric

1. On the **Custom Metrics** tab of the configuration tab of the logical aggregate table, find the custom metric that you want to delete and click **Delete** in the **Actions** column.
2. In the message that appears, click **OK**.
3. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
4. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

Move a custom metric to a specified category

1. On the **Custom Metrics** tab of the configuration tab of the logical aggregate table, find the custom metric that you want to move and click **Category** in the **Actions** column.
2. In the **Categorize Metric** dialog box, enter a group name.
3. Click **Create** to create a group.
4. Set the Categorize To parameter to the created group and click **OK**.
5. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to submit the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
6. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

9.9.1.9.3. Configure a logical table conversion task

To configure the custom parameters for a logical aggregate table, you can configure a logical table conversion task for the table. This topic describes how to configure a logical table conversion task for a logical aggregate table.

Prerequisites

Logical aggregate tables are created. For more information, see [Create a logical aggregate table](#).

Context

The default retention period of a logical aggregate table is 36,000 days. You cannot modify the retention period.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Aggregate Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the icon. The Logical Aggregate Tables section appears.
3. In the **Logical Aggregate Tables** section, move the pointer over the icon next to the logical aggregate table for which you want to configure a logical table conversion task and select **Change**.
4. On the configuration tab of the logical aggregate table, click **Logical Table Conversion Task Settings** in the top navigation bar.
5. On the **Logical Table Conversion Task Settings** tab, configure the custom parameters. For more information, see [Create a logical aggregate table](#).
6. Submit the logical aggregate table.
 - i. On the configuration tab of the logical aggregate table, click the icon in the upper-right corner to save the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
7. (Optional) Publish the logical aggregate table.
 - o If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

9.9.1.9.4. View details of a logical aggregate table

A logical aggregate table stores statistics about a statistic granularity. This topic describes how to view details of a logical aggregate table.

Prerequisites

Logical aggregate tables are created. For more information, see [Create a logical aggregate table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Aggregate Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Aggregate Tables section appears.
3. In the **Logical Aggregate Tables** section, move the pointer over the  icon next to the logical aggregate table for which you want to view details and select **Change**.
4. On the configuration tab of the logical aggregate table, click **Table Information** in the top navigation bar. In the **Logical Aggregate Table Details** pane, you can perform the following operations:
 - o View the primary key in the **Primary Key Information** section.
 - o View the name, display name, and description in the **Logical Aggregate Table Description** section.
 - o Click the dimension name next to **Granularity** to view the dimension information.Click **OK**. You can modify the description of the logical aggregate table in the **Logical Aggregate Table Description** section.
5. (Optional) Submit and publish the logical aggregate table. If you modify the description of the logical aggregate table in the **Logical Aggregate Table Description** section, submit and publish the logical aggregate table. To submit and publish a logical aggregate table, perform the following steps:
 - i. On the configuration tab of the logical aggregate table, click the  icon in the upper-right corner to save the logical aggregate table.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
 - iv. Publish the logical aggregate table.
 - If the current project is in Dev mode, publish the logical aggregate table to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the logical aggregate table after you submit it.

9.9.1.9.5. Delete a logical aggregate table

A logical aggregate table stores statistics about a statistic granularity. This topic describes how to delete a logical aggregate table.

Prerequisites

Logical aggregate tables are created. For more information, see [Create a logical aggregate table](#).

Context

You can delete only the logical aggregate tables that have no derived metrics or custom metrics.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Aggregate Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. Do not select the **Data_distill** project.
 - iii. On the **Develop** page, click the **Standards and Modeling** tab in the left-side navigation pane.
 - iv. On the **Standards and Modeling** tab, move the pointer over the left-side navigation submenu and click the  icon. The Logical Aggregate Tables section appears.
3. In the Logical Aggregate Tables section, move the pointer over the  icon next to the logical aggregate table that you want to delete and click **Delete**.
 4. In the **Tip** dialog box, enter your comments.
 5. Click **OK**.

9.9.1.10. Standards and modeling: Modeling engine

When logical tables are created and submitted, and relevant tasks are scheduled and run, Dataphin verifies logical table models, converts these logical tables to physical tables, and queries data in these logical tables based on the rules of the modeling engine. The intelligent modeling engine controls the model quality in the R&D process to ensure that logical table models are standard, reliable, stable, and efficient.

Query data in logical tables by using SQL statements

Dataphin allows you to query data in a logical table with the snowflake schema. Dataphin allows you to query data based on [Logical model.Dimension-associated field.Dimension-associated field....Attribute], such as Order.Buyer.Membership type.Type. This can improve the coding efficiency of SQL queries.

Query data in a logical table by using the following SQL syntax:

```
select
  LTable.Column, // Logical table name.Field name.
  LTable.Role(relation - dimension)...Dimension_Column, // Logical table name.Role name (Dimension-associated field name).Field name in the logical dimension table.
from LD.LTable // Business unit name.Logical table name.
where ...
```

Query data by using an alias:

```
select
  Alias.Column, // Alias. Field name.
  Alias.Role(relation - dimension)...Dimension_Column // Alias.Role name (Dimension-associated field name).Field name in the logical dimension table.
from LD.LTable Alias // Business unit name.Logical table name Alias.
where ...
```

Note

- If two dimension-associated fields that are referenced by dimension roles in a logical dimension table have the same name, you must set different aliases to avoid a conflict between field names in SQL statements.
- You can query data only in logical dimension tables of parent dimensions, including parent dimensions that are associated with fields of other tables and have dimension role names defined. If a logical dimension table is created based on a child dimension, you cannot query data in this table.
- Fields that are contained in the logical dimension table of a child dimension are included in the field list of the logical dimension table of the parent dimension.
- To reference a dimension that composes the statistic granularity of a logical aggregate table, you can associate the dimension with the logical aggregate table and execute the `SELECT * statement` on the logical aggregate table to obtain the primary key information of the logical dimension table. To prevent a slow full table scan, we recommend that you use the `SELECT Logical aggregate table.Logical dimension table.Primary key statement` to obtain the primary key information or use the `SELECT Logical aggregate table.Logical dimension table.Attribute statement` to obtain the attribute information.

Verify computing logic to prevent repetition

One of the core benefits of Dataphin is the unambiguous definition, which requires that the name and computing logic of each object be unique. When you submit a data standardization object or a logical table, Dataphin verifies the name, display name, and computing logic of the object. If duplicate computing logic exists, Dataphin displays a message to prevent you from creating objects with the same name but different definitions or objects with the same definition but different names.

When you create, modify, submit, or publish objects, Dataphin parses your requests based on the abstract syntax tree (AST) and checks whether duplicate computing logic and expressions exist in a namespace. Such objects include statistical periods, dimensions, logical dimension table fields, logical fact tables, logical fact table fields, atomic metrics, business filters, and custom metrics of logical aggregate tables. If duplicate computing logic exists, Dataphin displays a message to remind you.

Dynamically refresh model versions

Model versions are dynamically refreshed. This allows you to submit and convert logical tables more efficiently and modify logical tables more flexibly. In addition, Dataphin consumes fewer computing engine resources for modifying physical tables and modifying and migrating historical data. To obtain the latest data as needed, Dataphin schedules SQL tasks based on the latest physical table of a logical table.

To view the historical versions of a logical table, click the **Versions** tab in the upper part of the table configuration tab. For more information, see [View historical version information](#).

9.9.2. Batch processing

9.9.2.1. Overview

Dataphin allows you to create and run batch processing tasks during the data development process and manage the functions and files that are used in these tasks.

Dataphin allows you to create batch processing tasks of the `MAX_COMPUTE_SQL`, `MAX_COMPUTE_MR`, `SPARK_JAR_ON_MAX_COMPUTE`, `SHELL`, `PYTHON`, and `VIRTUAL` types. You can synchronize data, create destination tables, create complex data models, and process logical tables by using batch processing tasks.

The development process of a batch processing task consists of the following steps:

1. Create resources. For more information, see [Create a resource](#).
2. Create batch processing functions. For more information, see [Create a UDF](#).
3. Create a batch processing task. For more information, see [Create batch processing tasks](#).

9.9.2.2. Manage resources

9.9.2.2.1. Create a folder

This topic describes how to create a folder for storing resources.

Context

Folders help you categorize and manage resources.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. In the **Resource Management** section, click the  icon next to **Resource Management**.
7. In the **Create Folder** dialog box, enter a folder name and select a directory.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> In the Resource Management section, move the pointer over the  icon next to the folder that you want to rename and select Rename. In the field that appears, enter a new name. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> In the Resource Management section, move the pointer over the  icon next to the folder that you want to move and select Move. In the Move Folder dialog box, set the Select Directory parameter to the destination directory. Click OK.

Operation	Description
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none">i. In the Resource Management section, move the pointer over the  icon next to the folder that you want to delete.ii. Select Delete. <p> Note You can delete only the folders that do not contain subfolders or items.</p>

9.9.2.2.2. Create a resource

Dataphin provides the Resource Management section on the Develop page for you to store and manage files that are required during coding, such as JAR, JSON, and Python files. This topic describes how to create a resource.

Context

To reference files based on your business needs when you develop batch processing tasks, you can upload these files to Dataphin.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.

 **Note** If the current project is in Dev mode, publish the resource to the corresponding project in Prod mode. Then, you can view published resources in the production environment.

4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. Open the **Create Resource** dialog box by using one of the following methods:
 - In the Resource Management section, click the  icon next to **Resource Management**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Resources**.
 - In the right-side workspace of the **Develop** page, click the  icon next to **Resources**.
7. In the **Create Resource** dialog box, set the parameters as required.

Parameter	Description
Type	The type of the file to upload. Valid values: file , jar , Python , and others . The maximum size of each file is 50 MB. You can upload up to 1,000 files. <div style="border: 1px solid #add8e6; padding: 5px; background-color: #e6f2ff;"> <p>Note If the network connection is unstable, we recommend that you upload files that are not larger than 100 MB.</p> </div>
Name	The name of the resource to create. The name of the resource must be unique in the project.
Description	The description of the resource.
Upload File	The file to upload. Select the file to upload based on the value of the Type parameter.
Resource Usage	The usage of the resource. Valid values: Non-UDF , Batch Processing UDF , and Stream Processing UDF .
Select Directory	The directory for storing the resource. Select an existing directory.

8. Submit the resource.
 - i. Click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
9. (Optional) Publish the resource.
 - o If the current project is in **Dev** mode, publish the resource to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - o If the current project is in **Basic** mode, you do not need to publish the resource after you submit it.

9.9.2.2.3. Modify a resource

To modify an uploaded file, you can modify the corresponding resource and upload the modified file to replace the original file. This topic describes how to modify a resource.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional)On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to modify and select **Change**.
7. In the **Change Resource** dialog box, modify the parameters as needed. You can change the value of the **Description** parameter and re-upload a file.
8. Submit the resource.
 - i. Click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
9. (Optional)Publish the resource.
 - o If the current project is in Dev mode, publish the resource to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the resource after you submit it.

9.9.2.2.4. Move a resource

This topic describes how to move a resource to another directory.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional)On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination directory.
8. Click **OK**.

9.9.2.2.5. Delete a resource

You can delete a resource that is no longer used. This topic describes how to delete a resource.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to delete and select **Delete**.

 **Note** You cannot delete the resource if it is referenced by a task.

7. In the **Tip** dialog box, enter your comments.
8. Click **OK**.

9.9.2.3. Manage batch processing functions

9.9.2.3.1. Create a folder

This topic describes how to create a folder for storing functions.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Functions** section appears.
6. In the **Batch Processing Functions** section, click the  icon next to **Batch Processing Functions**.
7. In the **Create Folder** dialog box, enter a folder name and select a directory.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
-----------	-------------

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Batch Processing Functions section, move the pointer over the  icon next to the folder that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Batch Processing Functions section, move the pointer over the  icon next to the folder that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination directory. iii. Click OK.
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Batch Processing Functions section, move the pointer over the  icon next to the folder that you want to delete. ii. Select Delete. <div style="background-color: #e1f5fe; padding: 5px; margin-top: 10px;"> <p> Note You can delete only the folders that do not contain subfolders or items.</p> </div>

9.9.2.3.2. Create a UDF

Dataphin provides the Batch Processing Functions section on the Develop page. In this section, you can manage SQL functions to be used in batch processing tasks, including system built-in functions that are commonly used by computing engines and user-defined functions (UDFs). System built-in functions cannot be modified. This topic describes how to create a UDF.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Functions** section appears.
6. Open the **Create Function** dialog box by using one of the following methods:

- In the Batch Processing Functions section, click the  icon next to **Batch Processing Functions**.
- In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Batch Processing Functions**.
- In the right-side workspace of the **Develop** page, click the  icon below **Batch Processing Function**.

7. In the **Create Function** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the function to create. The name can contain letters, digits, and underscores (_), and must start with a letter.
Resource	<p>The resource to be referenced by the function. You can view the resources that are available for the current project in the drop-down list.</p> <div style="background-color: #e6f2ff; padding: 10px; border: 1px solid #d9e1f2;"> <p> Note</p> <ul style="list-style-type: none"> ○ You can create only functions that reference Python or JAR files. ○ If you select multiple resources, they must be of the same type. ○ If you do not have any resources, you must create one. For more information, see Create a resource. </div>
Class	The class of the function to create. You can extract the class from the resources in the MaxCompute computing engine. Example: <code>test_udf.UDFGETSrcId</code> .
Type	The type of the function to create. Valid values: Window , Statistic , Numeric , String , DateTime , ip address-related function , URL , Codec , Business , and Other .
Function Syntax	The syntax of the function to create. The svntax of a function is the format for referencing the function. Example: <code>bigintweekday (datetime date)</code> .
Description	<p>The description of the function to create.</p> <pre style="background-color: #f0f0f0; padding: 10px; border: 1px solid #d9d9d9;">select get_week_date("20170810",0,2), -- Query the date of the Tuesday in the week of August 10, 2017. from cndata.dual</pre>
Select Directory	The directory of the function to create. By default, the User-Defined Function directory is selected. You can also select another directory from the drop-down list.

8. Submit the UDF.
 - i. After you set the parameters, click **Submit** .
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**. The UDF is created and registered with MaxCompute.

 **Note** If the resources referenced by UDFs are updated, submit the UDFs again so that the UDFs registered with MaxCompute are updated.

You can reference the UDF in an SQL statement and execute the statement in a task on the **Ad Hoc Query** tab to check whether the function meets your expectation. For more information, see [Create an ad hoc query task](#). The following SQL statement is used as an example:

```
select
get_week_date("20170810",0,2), -- Query the date of the Tuesday in the week of August 10, 2017.
from cndata.dual
```

9. (Optional) Publish the UDF.
 - If the current project is in **Dev** mode, publish the UDF to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the UDF after you submit it.

9.9.2.3.3. Modify a UDF

Dataphin provides the Batch Processing Functions section on the Develop page. In this section, you can manage SQL functions to be used in batch processing tasks, including system built-in functions that are commonly used by computing engines and UDFs. System built-in functions cannot be modified. This topic describes how to modify a UDF.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Functions** section appears.
6. In the **Batch Processing Functions** section, move the pointer over the  icon next to the UDF that you want to modify and select **Change**.
7. In the **Change Function** dialog box, modify the parameters as needed. You can change the values of the **Resource**, **Class**, **Type**, **Function Syntax**, and **Description** parameters. For more information, see [Create a UDF](#).
8. Submit the UDF.
 - i. After you modify the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
9. (Optional) Publish the UDF.
 - If the current project is in **Dev** mode, publish the UDF to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the UDF. After you submit a UDF, it can be referenced in batch processing tasks that are scheduled.

9.9.2.3.4. Move a UDF

This topic describes how to move a UDF to another directory.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Functions** section appears.
6. In the **Batch Processing Functions** section, move the pointer over the  icon next to the UDF that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination directory.
8. Click **OK**.

9.9.2.3.5. Delete a UDF

Dataphin provides the Batch Processing Functions section on the Develop page. In this section, you can manage SQL functions to be used in batch processing tasks, including system built-in functions that are commonly used by computing engines and UDFs. System built-in functions cannot be modified. This topic describes how to delete a UDF.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Functions** section appears.
6. In the **Batch Processing Functions** section, move the pointer over the  icon next to the UDF that you want to delete and select **Delete**.

 **Note** You cannot delete a UDF that is referenced in a task.

7. In the Tip dialog box, enter your comments.
8. Click OK.

9.9.2.4. Create a folder for storing batch processing tasks

This topic describes how to create a folder for storing batch processing tasks.

Procedure

1. Log on to the Dataphin console.
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
4. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Tasks** section appears.
5. In the Batch Processing Tasks section, click the  icon next to **Batch Processing Tasks**.
6. In the **Create Folder** dialog box, enter a folder name and select a directory.
7. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	To rename a folder, perform the following steps: <ol style="list-style-type: none">i. In the Batch Processing Tasks section, move the pointer over the  icon next to the folder that you want to rename and select Rename.ii. In the field that appears, enter a new name.iii. Press the Enter key.
Move a folder	To move a folder, perform the following steps: <ol style="list-style-type: none">i. In the Batch Processing Tasks section, move the pointer over the  icon next to the folder that you want to move and select Move.ii. In the Move Folder dialog box, set the Select Directory parameter to the destination directory.iii. Click OK.
Delete a folder	To delete a folder, perform the following steps: <ol style="list-style-type: none">i. In the Batch Processing Tasks section, move the pointer over the  icon next to the folder that you want to delete.ii. Select Delete. <div style="background-color: #e0f2f7; padding: 5px;"><p> Note You can delete only the folders that do not contain subfolders or items.</p></div>

9.9.2.5. Create batch processing tasks

9.9.2.5.1. Create a batch processing task of the MAX_COMPUTE_SQL type

This topic describes how to create a batch processing task of the MAX_COMPUTE_SQL type in Dataphin.

Context

You can create a batch processing task of the MAX_COMPUTE_SQL type based on your needs to process existing data and obtain required data.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Tasks** section appears.
3. In the Batch Processing Tasks section, click the  icon next to **Batch Processing Tasks** and select **MAX_COMPUTE_SQL**. The Create Item dialog box appears.
4. Set the task parameters, write the code of the task, and then run the code.
 - i. In the **Create Item** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the batch processing task, for example, SQL.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.
 - iii. On the **Code Editor** tab, write the SQL code of the task.
 - iv. Click **Run** in the upper-right corner to run the code.
5. (Optional) Configure the scheduling policy.
 - o If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
6. Save and submit the task.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.

- ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
7. (Optional) Publish the task.
- o If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.5.2. Create a batch processing task of the MAX_COMPUTE_MR type

This topic describes how to create a batch processing task of the MAX_COMPUTE_MR type in Dataphin.

Prerequisites

JAR packages are uploaded. For more information, see [Create a resource](#).

Context

A batch processing task of the MAX_COMPUTE_MR type must reference JAR packages. Before you create a batch processing task of the MAX_COMPUTE_MR type, you must upload the JAR packages to be referenced by the task.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing Tab, move the pointer over the left-side navigation submenu and click the  icon. The **Batch Processing Tasks** section appears.
3. In the Batch Processing Tasks section, click the  icon next to **Batch Processing Tasks** and select **MAX_COMPUTE_MR**. The Create Item dialog box appears.
4. Set the task parameters, write the code of the task, and then run the code.
 - i. In the **Create Item** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the batch processing task, for example, MR.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node.
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.

iii. On the **Code Editor** tab, write the code of the task. In this example, write the following code:

```
@resource_reference{"mr_odps.jar"}
add jar mr_odps.jar as momo.jar -f;
jar -resources momo.jar -classpath mr_odps.jar hive.WordCountOdps wc_in wc_out;
```

iv. Click **Run** in the upper-right corner to run the code.

5. (Optional) Configure the scheduling policy.

- If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
- If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.

6. Save and submit the task.

- i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.
- ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.

7. (Optional) Publish the task.

- If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
- If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.5.3. Create a batch processing task of the SPARK_JAR_ON_MAX_COMPUTE type

This topic describes how to create a batch processing task of the SPARK_JAR_ON_MAX_COMPUTE type in Dataphin.

Prerequisites

JAR or Python packages are uploaded. For more information, see [Create a resource](#).

Context

A batch processing task of the SPARK_JAR_ON_MAX_COMPUTE type must reference JAR or Python packages. Before you create a batch processing task of the SPARK_JAR_ON_MAX_COMPUTE type, you must upload the JAR or Python packages to be referenced by the task.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the left-side navigation submenu, click the  icon.
3. In the **Batch Processing Tasks** section, click the  icon next to Batch Processing Tasks and select

SPARK_JAR_ON_MAX_COMPUTE.

4. Set the task parameters, write the code of the task, and then run the code.
 - i. In the **Create Item** dialog box, set the task parameters.

Parameter	Description
Name	The name of the batch processing task, for example, Spark_python.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.
 - iii. On the **Code Editor** tab, write the code of the task. In this example, write the following code:

```
@resource_reference{"spark.py"}
spark-submit
--deploy-mode cluster
--conf spark.hadoop.odps.task.major.version=cupid_v2
--conf spark.hadoop.odps.end.point=http://service.cn.maxcompute.aliyun.com/api
--conf spark.hadoop.odps.runtime.end.point=http://service.cn.maxcompute.aliyun-inc.com/api
--master yarn
spark.py
```

In the sample code, `required_resource{}` specifies the JAR or Python package to be referenced, and all the other statements are fixed.

- iv. Click **Run** in the upper-right corner to run the code.
5. (Optional) Configure the scheduling policy.
 - If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
6. Save and submit the task.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.
 - ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
7. (Optional) Publish the task.
 - If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.5.4. Create a batch processing task of the SHELL type

This topic describes how to create a batch processing task of the SHELL type in Dataphin.

Prerequisites

JSON packages are uploaded. For more information, see [Create a resource](#).

Context

A batch processing task of the SHELL type must reference JSON packages. Before you create a batch processing task of the SHELL type, you must upload the JSON packages to be referenced by the task.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the left-side navigation submenu, click the  icon.
3. In the **Batch Processing Tasks** section, click the  icon next to Batch Processing Tasks and select **SHELL**.
4. Set the task parameters, write the code of the task, and then run the code.
 - i. In the **Create Item** dialog box, set the task parameters.

Parameter	Description
Name	The name of the batch processing task, for example, DataX.
Schedule Type	The scheduling type of the task. In this example, select Recurring Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.
- iii. On the **Code Editor** tab, write the code of the task.

In this example, write the following code:

```
@required_resource{required_memory=2Gb;required_cpus=1.0}
@resource_reference{"Datax.json"}
python $DATA_HOME/bin/datax.py --jvm '-Xms2g -Xmx2g' Datax.json
```

In the sample code:

- `required_resource{}` specifies the computing resources to be allocated to the task.
- `$DATA_HOME` is a system built-in parameter that specifies the installation directory of the sync task. The entry class of the sync task is in the `bin` subdirectory of the installation directory.
- `--jvm '-Xms2g -Xmx2g'` specifies the memory size of the Java virtual machine (JVM) when a sync task is running. We recommend that you set the memory size the same as the value of `required_memory` in `required_resource`.

Enter the following code for a sync task that requires a small amount of computing resources:

```
@resource_reference{"Datax.json"}
python $DATA_HOME/bin/datax.py Datax.json
```

- iv. Click **Run** in the upper-right corner to run the code.
5. (Optional) Configure the scheduling policy.
 - o If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
6. Save and submit the task.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.
 - ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
7. (Optional) Publish the task.
 - o If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.5.5. Create a batch processing task of the PYTHON type

This topic describes how to create a batch processing task of the PYTHON type in Dataphin.

Prerequisites

Python packages are uploaded. For more information, see [Create a resource](#).

Context

A batch processing task of the PYTHON type must reference Python packages. Before you create a batch processing task of the PYTHON type, you must upload the Python packages to be referenced by the task.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the left-side navigation submenu, click the  icon.
3. In the **Batch Processing Tasks** section, click the  icon next to Batch Processing Tasks and select **PYTHON**.
4. Set the task parameters, write the code of the task, and then run the code.

- i. In the **Create Item** dialog box, set the task parameters.

Parameter	Description
Name	The name of the batch processing task, for example, Python.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.
- iii. On the Code Editor tab, write the code of the task.

In this example, write the following code:

```
#!/usr/bin/python
#-*- coding: latin-1 -*-
import time
import datetime
import base64
import hashlib
import json
import sys
import csv
from odps import ODPS
def main():
    print "Hello World"
```

We recommend that you comment out the first two lines of the Python code. This helps avoid code execution errors when the system encoding format is used.

- iv. Click **Run** in the upper-right corner to run the code.
5. (Optional) Configure the scheduling policy.
 - If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
 6. Save and submit the task.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.
 - ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
 7. (Optional) Publish the task.
 - If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.5.6. Create a zero load node of the VIRTUAL type

When you configure the scheduling policy for a node, you can configure the node to depend on the zero load node as required. This topic describes how to create a zero load node of the VIRTUAL type in Dataphin.

Context

Before you build a data model, you can create a zero load node for the data model. After that, when you configure the scheduling policy for a node in the data model, you can configure the node to depend on the zero load node.

Procedure

1. Log on to the Dataphin console.
2. Go to the **Batch Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the left-side navigation submenu, click the .
3. In the **Batch Processing Tasks** section, click the  icon next to Batch Processing Tasks and select **VIRTUAL**.
4. Set the task parameters, write the code of the task, and then run the code.
 - i. In the **Create Item** dialog box, set the task parameters.

Parameter	Description
Name	The name of the batch processing task, for example, Virtual.
Schedule Type	The scheduling type of the task. Valid values: Recurring Node and One-Time Node .
Description	The description of the task.
Select Directory	The folder for storing the task.

- ii. Click **OK**.
- iii. On the **Code Editor** tab, write the SQL code of the task.
- iv. Click **Run** in the upper-right corner to run the code.

 **Note** Dataphin does not run the code of a zero load node. Instead, Dataphin directly sets the code execution result to successful.

5. (Optional) Configure the scheduling policy.
 - o If you set the Schedule Type parameter to **Recurring Node**, you must configure the scheduling policy. For more information, see [Configure a scheduling policy](#).
 - If you set the Schedule Type parameter to **One-Time Node**, you do not need to configure the scheduling policy.
6. Save and submit the zero load node.
 - i. Click the  icon in the upper-right corner of the Code Editor tab to save the task.

- ii. Click the  icon in the upper-right corner of the Code Editor tab to submit the task.
7. (Optional) Publish the zero load node.
 - o If the current project is in Dev mode, publish the node to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - o If the current project is in Basic mode, you do not need to publish the node. The node can be scheduled in the production environment after you submit it.

9.9.2.6. Configure a scheduling policy

Dataphin allows you to configure the scheduling rules and dependencies of nodes to make sure that tasks can be properly scheduled. This topic describes how to configure a scheduling policy for a batch processing task.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Context

- Dataphin allows you to configure scheduling policies only for recurring tasks.
- A dependency is a semantic connection between two or more nodes. The status of an upstream node affects the running of its downstream nodes.
- After you configure dependencies for an upstream node and its downstream nodes, the downstream nodes can be run only after the upstream node is run. Before the system runs a node, the system checks the scheduling time that is configured for the node and determines whether to run the node.
- If a scheduling configuration is submitted before the specified scheduling time, the configuration takes effect after the specified scheduling time. If a dependency is configured after the specified scheduling time, the dependency takes effect for the instances that are generated the next day.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. Go to the **Scheduling Configuration** tab of the batch processing task for which you want to configure a scheduling policy.
 - i. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - ii. On the left-side navigation submenu, click .
 - iii. In the **Batch Processing Tasks** section, click the batch processing task for which you want to configure a scheduling policy. The Code Editor tab of the batch processing task appears.
 - iv. (Optional) Steal the lock of the batch processing task.
 - If the batch processing task is locked by yourself, you do not need to steal the lock.
 - If the batch processing task is locked by another user, click the  icon in the upper-right corner to steal the lock.
 - v. Click the **Scheduling Configuration** tab.
5. In the **Scheduling Configuration** pane, set the parameters as required.

- i. Set the parameters in the **Basic Information** section. Dataphin automatically generates the node name, node ID, node type, and owner. You cannot modify these parameters.

Parameter	Description
Description	The description of the scheduling policy.
Priority	<p>The priority based on which the batch processing task is scheduled. Valid values:</p> <ul style="list-style-type: none"> ▪ Lowest Priority ▪ Low Priority ▪ Medium Priority ▪ High Priority ▪ Highest Priority
Parameters	<p>The specified values for the parameters that are used in the code of the task. Dataphin allows you to specify the parameters that are used in the code of a task. The specified values are used when the task is run. You can click Parameters and Descriptions to know how to configure the parameters.</p>

- ii. Set the parameters in the **Scheduling Configuration** section.

Parameter	Description
Schedule Mode	<p>The scheduling mode of the task. Valid values:</p> <ul style="list-style-type: none"> ▪ Normal: runs the task based on the specified recurrence. By default, this option is selected. ▪ Dry-run: runs the task based on the specified recurrence. However, the scheduling system does not actually run the task but directly returns a success response. ▪ Pause Scheduling: runs the task based on the specified recurrence. However, the scheduling system does not actually run the task but directly returns a failure response. You can select this check box if you need to suspend a task and run it later.

Parameter	Description
<p>Recurrence</p>	<p>The recurrence of the task. Valid values:</p> <ul style="list-style-type: none"> Day: automatically runs the task once per day. When you create a recurring task, the task is set to run at 00:00 every day by default. You can also click the  icon to specify a time for the task to be run as needed. Week: automatically runs the task at a specified time point on specified days of each week. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not actually run the instance or consume resources but directly returns a success response. <p>Assume that you set Recurrence to Week and specify that the task is run every Monday and Tuesday. The scheduling system generates and runs instances every Monday and Tuesday. Every Wednesday, Thursday, Friday, Saturday, and Sunday, the scheduling system generates instances and returns success responses without running the instances.</p> Month: automatically runs the task at a specified time point on specified days of each month. On the other days, the scheduling system still generates an instance every day to ensure the proper running of downstream instances. However, the system does not actually run the instance or consume resources but directly returns a success response. <p>Assume that you set Recurrence to Month and specify that the task is run on the seventh day of each month. The scheduling system generates and runs an instance on the seventh day of each month. On the other days, the scheduling system generates instances and returns success responses without running the instances.</p> Hour: automatically runs the task at a specified interval during a specified time period or at specified time points every day. The scheduling system automatically generates instances for the task and runs the instances at the specified interval or time points. <p>Assume that you set Recurrence to Hour, select Time Period, and set the Start, End, and Interval parameters to 00:00, 23:59, and 1, respectively. The scheduling system automatically generates instances for the task and runs an instance every hour.</p> Minute: automatically runs the task at a specified interval during a specified time period every day. <p>Assume that you set Recurrence to Minute and set the Start, End, and Interval parameters to 00:00, 23:59, and 05, respectively. The scheduling system automatically generates instances for the task and runs an instance every 5 minutes.</p>
<p>Depend on Previous Instance</p>	<p>Specifies whether the current task is run after the previous instance of another task or of the current task is run. If you select Depend on Previous Instance, you must further select Current Task or Select Task.</p> <ul style="list-style-type: none"> If you select Current Task, the current task is run after the previous instance of the current task is run. If you select Select Task, enter a keyword in the search box that appears to search for and select one or more tasks to depend on.

iii. Set the parameters in the **Dependency** section.

Parameter	Description
<p>Start Parsing</p>	<p>Start Parsing is available for SQL tasks. After you click Start Parsing, Dataphin parses the table referenced in the code and finds the output name that is the same as the table name. Dataphin automatically configures the node corresponding to this output name as the upstream node on which the current node depends.</p> <p>If the project referenced in the code is a variable or no project is specified in the code, Dataphin parses the name of the production project by default to ensure stable scheduling. Assume that the name of the production project is <code>onedata_dev</code>.</p> <ul style="list-style-type: none"> ▪ If the code contains <code>select * from s_order</code>, the <code>onedata.s_order</code> node is parsed as the upstream node. ▪ If the code contains <code>select * from \${onedata}.s_order</code>, the <code>onedata.s_order</code> node is parsed as the upstream node. ▪ If the code contains <code>select * from onedata.s_order</code>, the <code>onedata.s_order</code> node is parsed as the upstream node. ▪ If the code contains <code>select * from onedata_dev.s_order</code>, the <code>onedata_dev.s_order</code> node is parsed as the upstream node.
<p>Upstream Dependency</p>	<p>The upstream nodes on which the current node depends. To specify an upstream node, perform the following steps:</p> <ol style="list-style-type: none"> a. Click Create Upstream Dependency. b. In the Create Upstream Dependency dialog box, search for an upstream node in one of the following ways: <ul style="list-style-type: none"> ▪ Enter a keyword to search for an upstream node whose output name contains the keyword. ▪ Enter virtual to search for the zero load node. A root node is generated for each tenant or enterprise during initialization. The root node is a zero load node and its name starts with virtual. <div style="border: 1px solid #add8e6; padding: 5px; margin: 10px 0;"> <p> Note The output name of a node is globally unique and is not case-sensitive.</p> </div> c. Click OK. <p>To remove a node from the upstream node list, click the  icon in the Actions column.</p>

Parameter	Description
Current Node	<p>The output name for the current node. You can set multiple output names for a node, which can be used to configure dependencies for other nodes. To set an output name, perform the following steps:</p> <ol style="list-style-type: none"> Click Add. In the Add Output Task Nodes for Current Task Node dialog box, enter an output name. Observe uniform rules when you set each output name for the current node. In general, set an output name in the format of <code>Project name.Table name</code>, which is not case-sensitive. This helps other users find this node when they configure the upstream dependency for their nodes. <p>For example, if the name of the development project is <code>onedata_dev</code>, we recommend that you set the output name to <code>onedata.s_order</code>. If you set the output name to <code>onedata_dev.s_order</code>, the current node can be parsed as an upstream node for other nodes only when the code contains <code>select * from onedata_dev.s_order</code>.</p> Click OK. <p>You can also perform the following operations on an existing output name:</p> <ul style="list-style-type: none"> To delete an output name, click the  icon in the Actions column. To view the downstream nodes of the current node, click the  icon in the Actions column.

6. Click **OK**.

9.9.2.7. Modify a batch processing task

Batch processing tasks are used to process data offline during data development. This topic describes how to modify a batch processing task.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Context

If the batch processing task that you want to modify is locked by another user, click the  icon in the upper-right corner to steal the lock.

Procedure

- Log on to the [Dataphin console](#).
- On the Dataphin homepage, click **R&D** in the top navigation bar.
- (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
- On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
- On the left-side navigation submenu, click .
- In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing

task that you want to modify and select **Change**.

7. Modify the batch processing task.
 - i. (Optional) Steal the lock of the batch processing task.
 - If the batch processing task is locked by yourself, you do not need to steal the lock.
 - If the batch processing task is locked by another user, click the  icon in the upper-right corner to steal the lock.
 - ii. Modify the code or scheduling policy of the batch processing task.
8. Save and submit the batch processing task.
 - i. Click the  icon in the upper-right corner to save the batch processing task.
 - ii. Click the  icon in the upper-right corner to submit the batch processing task.
 - iii. Publish the batch processing task.
 - If the current project is in Dev mode, publish the task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the task. The task can be scheduled in the production environment after you submit it.

9.9.2.8. Unpublish and delete a batch processing task

Batch processing tasks are used to process data offline during data development. This topic describes how to unpublish, unpublish and delete, and delete batch processing tasks in different states.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Context

- A batch processing task may be in the following states:
 - After you create and save a batch processing task, it enters the **Draft** state.
 - After you submit a batch processing task, it enters the **Submitted** state.
 - After you modify and save a batch processing task in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a batch processing task in the **Submitted** state, it enters the **Draft** state.
- Limits on unpublishing, deleting, and unpublishing and deleting batch processing tasks:
 - You can unpublish batch processing tasks only when they are in the **Developing** or **Submitted** state.
 - You can delete batch processing tasks only when they are in the **Draft** state.
 - You can unpublish and delete batch processing tasks only when they are in the **Developing** or **Submitted** state.

Unpublish a batch processing task

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.

5. On the left-side navigation submenu, click the  icon.
6. In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing task that you want to unpublish and select **Unpublish**.
7. In the **Tip** dialog box, enter your comments.
8. Click **OK**.

Unpublish and delete a batch processing task

1. In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing task that you want to unpublish and delete and select **Unpublish and Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

Delete a batch processing task

1. In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing task that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.2.9. View the version information about a batch processing task

This topic describes how to view the version information about a batch processing task, including the details and code of each version.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Batch Processing Tasks** section, click the batch processing task whose version information you want to view.
7. On the configuration tab of the task, click **Versions** in the top navigation bar.
8. In the **Versions** pane, view the historical versions of the batch processing task.
 - Click **Details** in the **Actions** column of a version to view the details about the version.
 - Click **Compare Code Version** in the **Actions** column of a version to compare the code of the version with that of other versions.

- Click **View Code** in the **Actions** column of a version to view the code of the version.

9.9.2.10. Move a batch processing task

Dataphin allows you to move a batch processing task to a specified folder. This topic describes how to move a batch processing task.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing task that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
8. Click **OK**.

9.9.2.11. Rename a batch processing task

Dataphin allows you to rename a batch processing task. This topic describes how to rename a batch processing task.

Prerequisites

Batch processing tasks are created. For more information, see [Create a batch processing task](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Batch Processing Tasks** section, move the pointer over the  icon next to the batch processing task that you want to rename and select **Rename**.
7. In the field that appears, enter a new name.
8. Press the Enter key.

9.9.3. Stream processing

9.9.3.1. Overview

Data Processing allows you to run and manage stream processing tasks that are created during the data development process. You can also manage the files, metatables, and templates that are used in these tasks.

Dataphin allows you to create stream processing tasks of the `FLINK_SQL` and `FLINK_TEMPLATE_SQL` types. For more information, see [Create a stream processing task of the FLINK_SQL type](#) and [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

To develop a stream processing task, perform the following steps:

1. Create a resource. For more information, see [Create a resource](#).
2. Create a stream processing function. For more information, see [Create a UDF](#).
3. Create a stream metatable. For more information, see [Create a stream metatable](#).
4. Optional. Create a stream processing template. For more information, see [Create a stream processing template](#).
5. Create a stream processing task. For more information, see [Create a stream processing task of the FLINK_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

 **Note** Stream processing templates can be used only by stream processing tasks of the `FLINK_TEMPLATE_SQL` type.

9.9.3.2. Manage resources

9.9.3.2.1. Create a folder

This topic describes how to create a folder for storing resources.

Context

Folders help you categorize and manage resources.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. In the **Resource Management** section, click the  icon next to **Resource Management**.
7. In the **Create Folder** dialog box, enter a folder name and select a directory.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
-----------	-------------

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Resource Management section, move the pointer over the  icon next to the folder that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Resource Management section, move the pointer over the  icon next to the folder that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination directory. iii. Click OK.
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Resource Management section, move the pointer over the  icon next to the folder that you want to delete. ii. Select Delete. <div style="background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p> Note You can delete only the folders that do not contain subfolders or items.</p> </div>

9.9.3.2.2. Create a resource

Dataphin provides the Resource Management section on the Develop page for you to store and manage files that are required during coding, such as JAR, JSON, and Python files. This topic describes how to create a resource.

Context

To reference files based on your business needs when you develop batch processing tasks, you can upload these files to Dataphin.

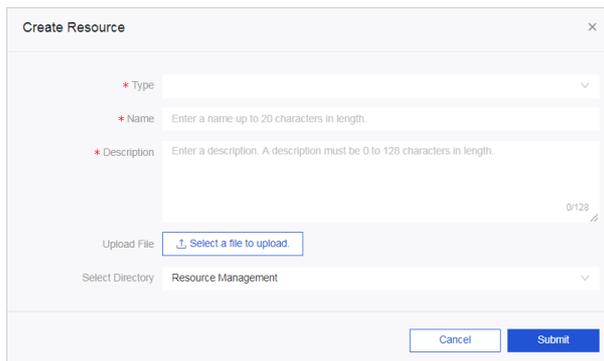
Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.

 **Note** If the current project is in Dev mode, publish the resource to the corresponding project in Prod mode. Then, you can view published resources in the production environment.

4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.

5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. Open the **Create Resource** dialog box by using one of the following methods:
 - In the Resource Management section, click the  icon next to **Resource Management**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Resources**.
 - In the right-side workspace of the **Develop** page, click the  icon next to **Resources**.
7. In the **Create Resource** dialog box, set the parameters as required.



Parameter	Description
Type	The type of the file to upload. Valid values: file , jar , Python , and others . The maximum size of each file is 50 MB. You can upload up to 1,000 files. Note If the network connection is unstable, we recommend that you upload files that are not larger than 100 MB.
Name	The name of the resource to create. The name of the resource must be unique in the project.
Description	The description of the resource.
Upload File	The file to upload. Select the file to upload based on the value of the Type parameter.
Resource Usage	The usage of the resource. Valid values: Non-UDF , Batch Processing UDF , and Stream Processing UDF .
Select Directory	The directory for storing the resource. Select an existing directory.

8. Submit the resource.
 - i. Click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
9. (Optional) Publish the resource.
 - If the current project is in **Dev** mode, publish the resource to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).

- If the current project is in **Basic** mode, you do not need to publish the resource after you submit it.

9.9.3.2.3. Modify a resource

To modify an uploaded file, you can modify the corresponding resource and upload the modified file to replace the original file. This topic describes how to modify a resource.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to modify and select **Change**.
7. In the **Change Resource** dialog box, modify the parameters as needed. You can change the value of the **Description** parameter and re-upload a file.
8. Submit the resource.
 - i. Click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
9. (Optional) Publish the resource.
 - If the current project is in **Dev** mode, publish the resource to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the resource after you submit it.

9.9.3.2.4. Move a resource

This topic describes how to move a resource to another directory.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon.

The **Resource Management** section appears.

6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination directory.
8. Click **OK**.

9.9.3.2.5. Delete a resource

You can delete a resource that is no longer used. This topic describes how to delete a resource.

Prerequisites

Resources are created. For more information, see [Create a resource](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Resource Management** section appears.
6. In the **Resource Management** section, move the pointer over the  icon next to the resource that you want to delete and select **Delete**.

 **Note** You cannot delete the resource if it is referenced by a task.

7. In the **Tip** dialog box, enter your comments.
8. Click **OK**.

9.9.3.3. Manage stream processing functions

9.9.3.3.1. Create a folder for storing stream processing functions

This topic describes how to create a folder for storing stream processing functions.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.

5. On the left-side navigation submenu, click the  icon.
6. In the **Stream Processing Functions** section, click the  icon next to Stream Processing Functions.
7. In the **Create Folder** dialog box, enter a folder name and select a folder.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination folder. iii. Click OK.
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to delete. ii. Select Delete. <div style="background-color: #e0f2f1; padding: 5px; margin-top: 10px;"> <p> Note You can delete only the folders that do not contain subfolders or items.</p> </div>

9.9.3.3.2. Create a UDF

Dataphin provides the Stream Processing Functions section on the Develop page. In this section, you can manage functions to be used in stream processing tasks, including system built-in functions and UDFs. System built-in functions cannot be modified. This topic describes how to create a UDF.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click .
6. Open the **Create Function** dialog box in one of the following ways:

- In the Stream Processing Functions section, click the  icon next to **Stream Processing Functions**.
- In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Stream Processing Functions**.
- In the right-side workspace of the **Develop** page, click the  icon next to **Stream Processing Functions**.

7. In the **Create Function** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the function to create. The name can contain letters, digits, and underscores (_), and must start with a letter.
Resource	<p>The resource to be referenced by the function. You can view the resources that are available for the current project in the drop-down list.</p> <div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p> Note</p> <ul style="list-style-type: none"> ○ If you select multiple resources, they must be of the same type. ○ If you do not have any resources, you must create one. For more information, see Create a resource. </div>
Class	The class of the function to create. You can extract the class from the resources in the MaxCompute computing engine. Example: <code>test_udf.UDFGETSrcId</code> .
Type	The type of the function to create. Valid values: Window, Statistic, Numeric, String, DateTime, ip address-related function, URL, Codec, Business, and Other .
Function Syntax	The syntax of the function to create. The syntax of a function is the format for referencing the function. Example: <code>bigintweekday (datetime date)</code> .
Description	<p>The description of the function to create. Example:</p> <pre style="background-color: #f5f5f5; padding: 10px; border: 1px solid #ccc;">select get_week_date("20170810",0,2), -- Query the date of the Tuesday in the week of August 10, 2017. from cndata.dual</pre>
Select Directory	The folder of the function to create.

8. Submit the UDF.

- i. After you set the parameters, click **Submit**.
- ii. In the dialog box that appears, enter your comments.
- iii. Click **OK**. The UDF is created and registered with MaxCompute.

 **Note** If the resources referenced by UDFs are updated, submit the UDFs again so that the UDFs registered with MaxCompute are updated.

You can reference the UDF in an SQL statement and execute the statement in a task on the Ad Hoc Query tab to check whether the function meets your expectation. The following SQL statement is used as an example:

```
select
get_week_date("20170810",0,2), -- Query the date of the Tuesday in the week of August 10, 2017.
from cndata.dual
```

9. (Optional) Publish the UDF.
 - If the current project is in Dev mode, publish the UDF to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the UDF after you submit it.

9.9.3.3.3. Modify a UDF

Dataphin provides the Stream Processing Functions section on the Develop page. In this section, you can manage functions to be used in stream processing tasks, including system built-in functions and UDFs. System built-in functions cannot be modified. This topic describes how to modify a UDF.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Stream Processing Functions** section, move the pointer over the  icon next to the UDF that you want to modify and select **Change**.
7. In the **Change Function** dialog box, set the **Resource**, **Class**, **Type**, **Function Syntax**, and **Description** parameters. For more information about the parameter description, see [Create a UDF](#).
8. Submit the UDF.
 - i. After you modify the parameters, click **Submit**.
 - ii. In the dialog box that appears, enter your comments.
 - iii. Click **OK**.
 - iv. (Optional) Publish the UDF.
 - If the current project is in Dev mode, publish the UDF to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the UDF. After you submit a UDF, it can be referenced in stream processing tasks for scheduling.

9.9.3.3.4. Move a UDF

Dataphin provides the Stream Processing Functions section on the Develop page. In this section, you can manage functions to be used in stream processing tasks, including system built-in functions and UDFs. System built-in functions cannot be modified. This topic describes how to move a UDF.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional)On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode.You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Stream Processing Functions** section, move the pointer over the  icon next to the UDF that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
8. Click **OK**.

9.9.3.3.5. Delete a UDF

Dataphin provides the Stream Processing Functions section on the Develop page. In this section, you can manage functions to be used in stream processing tasks, including system built-in functions and UDFs. System built-in functions cannot be modified. This topic describes how to delete a UDF.

Prerequisites

UDFs are created. For more information, see [Create a UDF](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional)On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode.You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the left-side navigation submenu, click the  icon.
6. In the **Stream Processing Functions** section, move the pointer over the  icon next to the UDF that you want to delete and select **Delete**.

 **Note** You cannot delete a UDF that is referenced in a task.

7. In the **Tip** dialog box, enter your comments.
8. Click **OK**.

9.9.3.4. Manage stream processing metatables

9.9.3.4.1. Create a folder for storing stream metatables

This topic describes how to create a folder for storing stream metatables.

Procedure

1. Log on to the Dataphin console.
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
6. In the **Stream Metatables** section, click the  icon next to **Stream Metatables**.
7. In the **Create Folder** dialog box, enter a folder name and select a folder.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	To rename a folder, perform the following steps: <ol style="list-style-type: none">i. In the Stream Metatables section, move the pointer over the  icon next to the folder that you want to rename and select Rename.ii. In the field that appears, enter a new name.iii. Press the Enter key.
Move a folder	To move a folder, perform the following steps: <ol style="list-style-type: none">i. In the Stream Metatables section, move the pointer over the  icon next to the folder that you want to move and select Move.ii. In the Move Folder dialog box, set the Select Directory parameter to the destination folder.iii. Click OK.
Delete a folder	To delete a folder, perform the following steps: <ol style="list-style-type: none">i. In the Stream Metatables section, move the pointer over the  icon next to the folder that you want to delete.ii. Select Delete. <div style="background-color: #e0f2f7; padding: 5px;"><p> Note You can delete only the folders that do not contain subfolders or items.</p></div>

9.9.3.4.2. Create a stream metatable

Dataphin provides the Stream Metatables section on the Develop page. In this section, you can manage stream metatables to be used in stream processing tasks, including input tables, output tables, and dimension tables. This topic describes how to create a stream metatable.

Procedure

1. Log on to the Dataphin console.
2. Go to the Stream Metatables section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
3. Open the **Create Stream Metatable** dialog box by using one of the following methods:
 - In the **Stream Metatables** section, click the  icon next to Stream Metatables.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Stream Metatables**.
 - In the right-side workspace of the **Develop** page, click the  icon next to **Stream Metatables**.
4. In the **Create Stream Metatable** dialog box, set the parameters as required.

Parameter	Description
Metatable Name	The name of the stream metatable.  Note The name must be unique in the project. Otherwise, the stream metatable cannot be published to the production environment.
Data Source	The data source of the stream metatable. The data sources that are connected to Dataphin are displayed in the drop-down list.
Source Table	The source table. If you select a data source of the ALIYUN_HBASE , KAFKA_9_11 , TABLE_STORE , or DRDS type, enter the name of a source table from the corresponding data source as needed.
Topic	The source topic. If you select a data source of the DATAHUB or LOG_SERVICE type, select a topic from the drop-down list or enter the name of a topic from the corresponding data source.
Topic	The source topic. If you select a data source of the ROCKET_MQ type, enter the name of a topic from the corresponding data source.
Select Directory	The folder for storing the stream metatable.
Description	The description of the stream metatable.

5. Click **OK**.
6. (Optional) Add one or more fields to the stream metadata by using one of the following methods.
 - a. On the configuration tab of the stream metadata, move the pointer over **Add Field** and select **Use SQL Code**.
 - b. Enter code in the code editor of the **Use SQL Code** dialog box. Sample code:

```
create table dwi_pub_hbd_cate_mtr (
  rowkey VARCHAR comment 'rowkey',
  stat_date VARCHAR comment 'stat_date',
  keymin VARCHAR comment 'keymin',
  PRIMARY KEY(rowkey)
)
with (
  type='alibase',
  diamondGroup='null',
  zkQuorum='hbasemaster74000.sg94.tbsite.net,hbasemaster74001.sg94.tbsite.net,hbasemaster74002.sg94.tbsite.net,hbase74000.sg94.tbsite.net,hbase74001.sg94.tbsite.net',
  diamondKey='null',
  zkNodeParent='/group-sg94-lzd-mix',
  columnFamily='info',
  tableName='dwi_pub_hbd_cate_mtr_001',
  stringWriteMod='true'
);
```

- c. Click **OK**.
- a. On the configuration tab of the stream metadata, move the pointer over **Add Field** and select **Add Multiple Fields**.
- b. Enter code in the code editor of the **Add Multiple Fields** dialog box. Sample code:

```
ID, INT, description, false, true
name, INT, description, false, true
```

- a. On the configuration tab of the stream metadata, move the pointer over **Add Field** and select **Add Single Field**.
- b. In the **Add Single Field** dialog box, set the parameters as required.
- c.

Parameter	Description
Name	The name of the field.
Data Type	The data type of the field.
Primary Key	Specifies whether the field is a primary key.
Header	Specifies whether the field is an attribute field of the data source.
Description	The description of the field.

- d. Click **OK**.

The following table describes the operations that you can perform on added fields.

Operation	Description
Edit a field	To edit a field, perform the following steps: <ol style="list-style-type: none"> i. In the field list of the stream metatable configuration page, find the field that you want to edit and click the  icon in the Actions column. ii. In the Edit dialog box, modify the parameters as required. iii. Click Finish.
Delete a field	To delete a field, perform the following steps: <ol style="list-style-type: none"> i. In the field list of the stream metatable configuration page, find the field that you want to delete and click the  icon in the Actions column. ii. In the Information message, click OK.
Sort fields	To sort fields, perform the following steps: <ol style="list-style-type: none"> i. On the stream metatable configuration page, click Sort By. ii. Sort the fields and click Finish.
Search for fields	On the stream metatable configuration page, enter a name or a keyword in the search box to search for fields.
Refresh	On the stream metatable configuration page, click the  icon to reparse the fields from the stream metatable and refresh the field list.

7. Save and submit the stream metatable.

- i. Click the  icon in the upper-right corner of the configuration tab to save the stream metatable.
- ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream metatable.

8. (Optional) Publish the stream metatable.

- If the current project is in Dev mode, publish the stream metatable to the corresponding project in Prod mode. For more information, see [Publishing management](#).
- If the current project is in Basic mode, you do not need to publish the stream metatable after you submit it.

9.9.3.4.3. Modify a stream metatable

Dataphin provides the Stream Metatables section on the Develop page. In this section, you can manage stream metatables to be used in stream processing tasks, including input tables, output tables, and dimension tables. This topic describes how to modify a stream metatable.

Prerequisites

Stream metatables are created. For more information, see [Create a stream metatable](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Metatables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
3. In the **Stream Metatables** section, move the pointer over the  icon next to the stream metatable that you want to modify and select **Change**.
4. (Optional) Steal the lock of the stream metatable.
 - i. Steal the lock.
 - If the stream metatable is locked by yourself, you do not need to steal the lock.
 - If the stream metatable is locked by another user, click the  icon in the upper-right corner to steal the lock.
 - ii. After the lock is stolen, go to the next step to modify the metatable information and the fields in the stream metatable.
5. Modify the metatable information.
 - i. In the top navigation bar of the configuration tab, click **Metatable Information**.
 - ii. In the **Metatable Information** pane, modify the **DataSource**, **Source topic** or **Source table**, **Description**, and **Parameter k-v configuration** parameters.
 - iii. Click **OK**.
6. Modify the fields in the stream metatable. For more information, see [Create a stream metatable](#).
7. Save and submit the stream metatable.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream metatable.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream metatable.
8. (Optional) Publish the stream metatable.
 - If the current project is in **Dev** mode, publish the stream metatable to the corresponding project in **Prod** mode. For more information, see [Publishing management](#).
 - If the current project is in **Basic** mode, you do not need to publish the stream metatable after you submit it.

9.9.3.4.4. View the version information about a stream metatable

Dataphin provides the **Stream Metatables** section on the **Develop** page. In this section, you can manage stream metatables to be used in stream processing tasks, including input tables, output tables, and dimension tables. This topic describes how to view the version information about a stream metatable.

Prerequisites

Stream metatables are created. For more information, see [Create a stream metatable](#).

Procedure

1. [Log on to the Dataphin console](#).

2. Go to the **Stream Metatables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
3. In the Stream Metatables section, click the stream metatable for which you want to view the version information.
4. In the top navigation bar of the configuration tab that appears, click **Version Details**. The Version Details pane appears.
5. Click the  icon in the **Actions** column of a version to view the comparison between the current version and the reference version.
6. Click the  icon in the **Actions** column of a version to view the code of the version.

9.9.3.4.5. Move a stream metatable

Dataphin provides the Stream Metatables section on the Develop page. In this section, you can manage stream metatables to be used in stream processing tasks, including input tables, output tables, and dimension tables. This topic describes how to move a stream metatable.

Prerequisites

Stream metatables are created. For more information, see [Create a stream metatable](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Metatables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
3. In the **Stream Metatables** section, move the pointer over the  icon next to the stream metatable that you want to move and select **Move**.
4. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
5. Click **OK**.

9.9.3.4.6. Delete a stream metatable

Dataphin provides the Stream Metatables section on the Develop page. In this section, you can manage stream metatables to be used in stream processing tasks, including input tables, output tables, and dimension tables. This topic describes how to delete a stream metatable.

Prerequisites

Stream metatables are created. For more information, see [Create a stream metatable](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Metatables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Metatables** section appears.
3. In the **Stream Metatables** section, move the pointer over the  icon next to the stream metatable that you want to delete and select **Delete**.

 **Note** You cannot delete a stream metatable if it is referenced by a stream processing task.

4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

9.9.3.5. Manage stream processing templates

9.9.3.5.1. Create a folder for storing stream processing templates

This topic describes how to create a folder for storing stream processing templates.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.
6. In the **Stream Processing Templates** section, click the  icon next to Stream Processing Templates.
7. In the **Create Folder** dialog box, enter a folder name and select a folder.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination folder. iii. Click OK.
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Templates section, move the pointer over the  icon next to the folder that you want to delete. ii. Select Delete. <div style="background-color: #e0f2f1; padding: 5px; margin-top: 10px;"> <p> Note You can delete only the folders that do not contain subfolders or items.</p> </div>

9.9.3.5.2. Create a stream processing template

Dataphin provides the Stream Processing Templates section on the Develop page. In this section, you can manage templates that can be used in stream processing tasks to improve the coding efficiency. This topic describes how to create a stream processing template.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Stream Processing Templates** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.
3. Open the **Create Stream Processing Template** dialog box by using one of the following methods:
 - In the **Stream Processing Functions** section, click the  icon next to Stream Processing Templates.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Stream Processing Templates**.

- In the right-side workspace of the **Develop** page, click the  icon next to **Stream Processing Template**.
- 4. In the **Create Stream Processing Template** dialog box, set the **Template Name**, **Select Directory**, and **Description** parameters.
- 5. Click **OK**.
- 6. Configure the stream processing template.
 - i. Write the code of the stream processing template.
 - ii. In the top navigation bar of the configuration tab, click **Template Details**.
 - iii. In the **Template Details** pane, click **Parse** in the Parameter Configuration section.
 - iv. Set **Parameter Description** and **Default Value** for the parameters that are used in the stream processing template as required. In the Basic Information section, you can also modify or add a description for the template.
 - v. Click **OK**. If you reference this template when you create a stream processing task of the `FLINK_TEMPLATE_SQL` type, you can modify the parameters in the template on the configuration tab of the stream processing task.
- 7. Save and submit the stream processing template.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream processing template.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream processing template.
- 8. (Optional) Publish the stream processing template.
 - If the current project is in Dev mode, publish the stream processing template to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the stream processing template after you submit it.

9.9.3.5.3. Modify a stream processing template

Dataphin provides the Stream Processing Templates section on the Develop page. In this section, you can manage templates that can be used in stream processing tasks to improve the coding efficiency. This topic describes how to modify a stream processing template.

Prerequisites

Stream processing templates are created. For more information, see [Create a stream processing template](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Processing Templates** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.

3. In the **Stream Processing Templates** section, move the pointer over the  icon next to the stream processing template that you want to modify and select **Change**.
4. (Optional)Steal the lock of the stream processing template.
 - i. Steal the lock.
 - If the stream processing template is locked by yourself, you do not need to steal the lock.
 - If the stream processing template is locked by another user, click the  icon in the upper-right corner to steal the lock.
 - ii. After the lock is stolen, go to the next step to modify the code and parameters in the stream processing template
5. Modify the code and parameters in the stream processing template. For more information, see [Create a stream processing template](#).
6. Save and submit the stream processing template.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream processing template.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream processing template.
7. (Optional)Publish the stream processing template.
 - If the current project is in Dev mode, publish the stream processing template to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the stream processing template after you submit it.

9.9.3.5.4. View the version information about a stream processing template

Dataphin provides the Stream Processing Templates section on the Develop page. In this section, you can manage templates that can be used in stream processing tasks to improve the coding efficiency. This topic describes how to view the version information about a stream processing template.

Prerequisites

Stream processing templates are created. For more information, see [Create a stream processing template](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Processing Templates** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.
3. In the Stream Processing Templates section, click the stream processing template for which you want to

view the version information.

4. In the top navigation bar of the configuration tab that appears, click **Version Details**. The Version Details pane appears.
5. Click the  icon in the **Actions** column of a version to view the comparison between the current version and the reference version.
 - a. In the **Version Comparison** dialog box, click the **Code Comparison** tab to view the comparison between the code in the **Current Version** and **Reference Version** sections.
 - b. To roll back to the reference version, click **Roll Back to Reference Version**.
 - c. In the **Tip** message, click **OK**.
 - a. In the **Version Comparison** dialog box, click the **Template Parameter Comparison** tab to view the comparison between the parameter information in the **Current Version** and **Reference Version** sections.
 - b. To roll back to the reference version, click **Roll Back to Reference Version**.
 - c. In the **Tip** message, click **OK**.
6. Click the  icon in the **Actions** column of a version to view the code of the version.

9.9.3.5.5. Move a stream processing template

Dataphin provides the Stream Processing Templates section on the Develop page. In this section, you can manage templates that can be used in stream processing tasks to improve the coding efficiency. This topic describes how to move a stream processing template.

Prerequisites

Stream processing templates are created. For more information, see [Create a stream processing template](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Processing Templates** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.
3. In the **Stream Processing Templates** section, move the pointer over the  icon next to the stream processing template that you want to move and select **Move**.
4. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
5. Click **OK**.

9.9.3.5.6. Delete a stream processing template

Dataphin provides the Stream Processing Templates section on the Develop page. In this section, you can manage templates that can be used in stream processing tasks to improve the coding efficiency. This topic describes how to delete a stream processing template.

Prerequisites

Stream processing templates are created. For more information, see [Create a stream processing template](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Processing Templates** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Templates** section appears.
3. Move the pointer over the  icon next to the stream processing template that you want to delete and select **Delete**.

 **Note** You can delete only the folders that do not contain subfolders or items.

4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

9.9.3.6. Manage stream processing tasks

9.9.3.6.1. Create a folder for storing stream processing tasks

This topic describes how to create a folder for storing stream processing tasks.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
6. In the **Stream Processing Tasks** section, click the  icon next to Stream Processing Tasks.
7. In the **Create Folder** dialog box, enter a folder name and select a folder.
8. Click **OK**. The following table describes the operations that you can perform on created folders.

Operation	Description
Rename a folder	<p>To rename a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Tasks section, move the pointer over the  icon next to the folder that you want to rename and select Rename. ii. In the field that appears, enter a new name. iii. Press the Enter key.
Move a folder	<p>To move a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Tasks section, move the pointer over the  icon next to the folder that you want to move and select Move. ii. In the Move Folder dialog box, set the Select Directory parameter to the destination folder. iii. Click OK.
Delete a folder	<p>To delete a folder, perform the following steps:</p> <ol style="list-style-type: none"> i. In the Stream Processing Tasks section, move the pointer over the  icon next to the folder that you want to delete. ii. Select Delete. <div style="background-color: #e0f2f7; padding: 10px; margin-top: 10px;"> <p> Note You can delete only the folders that do not contain subfolders or items.</p> </div>

9.9.3.6.2. Create a stream processing task of the FLINK_SQL type

Dataphin allows you to manage stream processing tasks of the FLINK_SQL and FLINK_TEMPLATE_SQL types. This topic describes how to create a stream processing task of the FLINK_SQL type.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
3. Create a stream processing task of the FLINK_SQL type.

- i. Open the **Create Flink_SQL** dialog box by using one of the following methods:
 - In the **Stream Processing Tasks** section, click the  icon next to **Stream Processing Tasks** and select **FLINK_SQL**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Stream Processing Tasks > FLINK_SQL**.
 - In the right-side workspace of the **Develop** page, click the  icon next to **FLINK_SQL**.
- ii. In the **Create Flink_SQL** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the stream processing task. <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> ? Note The name must be unique in the project. Otherwise, the stream processing task cannot be published to the production environment. </div>
Select Directory	The folder for storing the stream processing task.
Resource Queue	The resource queue of the computing engine for stream processing that is bound to the current project.
Engine Version	The version of the computing engine supported by the selected resource queue.

iii. Click **OK**.

4. Write the code of the stream processing task.

- i. On the configuration tab, write the code to configure the stream processing task.

? **Note** If you do not have permissions to access the tables referenced in the stream processing task, enter related **SET** statements at the beginning of the code to specify the **AccessKey ID** and **AccessKey secret** required for accessing the tables.

- ii. Click **Precompile** to check the syntax and related permissions of the stream processing task.
 - iii. Click **Debugging**. Then, you can configure the sample data for debugging and locally debug the stream processing task to make sure that the code to be submitted is correct.
 - iv. In the **Configure Debugging Data** dialog box, select a table in the left-side table list and click **Auto Sampling**.
 - v. In the pop-up dialog box, set the **Data Records** parameter and click **OK**.
 - vi. After the sampling is complete for all tables, click **OK** in the lower-right corner of the **Configure Debugging Data** dialog box.
 - vii. View the debugging data, intermediate result, and debugging result on the **Result** tab.
5. Set the parameters of the stream processing task.
- i. In the top navigation bar of the configuration tab, click **Task Parameters**.
 - ii. In the **Task Parameters** pane, set parameters, including the checkpoint mode and interval, based on the sample code of the stream processing task.
 - iii. Click **OK**.

6. Configure the stream processing task.
 - i. In the top navigation bar of the configuration tab, click **Task Configuration**. The **Task Configuration** pane appears.
 - ii. Set the **Resource Configuration** parameter.
 - If you select **System Recommended**, you do not need to configure resources.
 - If you select **Custom**, you must click **Configure** to configure resources. You can configure resources in one of the following modes: **Visualization Mode** and **Code Editor**.

Mode	Description
Visualization Mode	To configure resources in visualization mode, perform the following steps: <ol style="list-style-type: none">a. On the resource configuration page, click Visualization Mode.b. Click the  icon in the upper-right corner of a compute node.c. In the dialog box that appears, set the parameters as required.d. Click OK.e. After you set the execution parameters for all compute nodes, click Save in the upper-right corner of the page.
Code Editor	To configure resources in the code editor, perform the following steps: <ol style="list-style-type: none">a. On the resource configuration page, click Code Editor.b. In the code editor, set the execution parameters for compute nodes as required. Then, click Save in the upper-right corner of the page.

- iii. In the **Task Configuration** pane, click the  icon to turn on **Auto Optimization** and set the **Maximum CUs** and **Expected Maximum Memory** parameters as required. Then, click **OK**.
7. Save and submit the stream processing task.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream processing task.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream processing task.
8. (Optional) Publish the stream processing task.
 - If the current project is in Dev mode, publish the stream processing task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the stream processing task after you submit it.

9.9.3.6.3. Create a stream processing task of the FLINK_TEMPLATE_SQL type

Dataphin allows you to manage stream processing tasks of the FLINK_SQL and FLINK_TEMPLATE_SQL types. This topic describes how to create a stream processing task of the FLINK_TEMPLATE_SQL type.

Prerequisites

Stream processing templates are created. For more information, see [Create a stream processing template](#).

Procedure

1. Log on to the Dataphin console.
2. Go to the **Stream Processing Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
 - iii. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
 - iv. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
3. Create a stream processing task of the **FLINK_TEMPLATE_SQL** type.
 - i. Open the **Create Flink_Template_SQL** dialog box by using one of the following methods:
 - In the **Stream Processing Tasks** section, click the  icon next to **Stream Processing Tasks** and select **FLINK_TEMPLATE_SQL**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Data Processing > Stream Processing Tasks > FLINK_TEMPLATE_SQL**.
 - In the right-side workspace of the **Develop** page, click the  icon next to **FLINK_TEMPLATE_SQL**.
 - ii. In the **Create Flink_Template_SQL** dialog box, set the parameters as required.

Parameter	Description
Name	The name of the stream processing task.  Note The name must be unique in the project. Otherwise, the stream processing task cannot be published to the production environment.
Select Directory	The folder for storing the stream processing task.
Resource Queue	The resource queue of the computing engine for stream processing that is bound to the current project.
Engine Version	The version of the computing engine supported by the selected resource queue.
Referenced Template	The template referenced by the stream processing task.

- iii. Click **OK**.
4. Write the code of the stream processing task.
 - i. On the configuration tab, write the code to configure the stream processing task.
 - ii. Click **Precompile** to check the syntax and related permissions of the stream processing task.
 - iii. Click **Debugging**. Then, you can configure the sample data for debugging and locally debug the stream processing task to make sure that the code to be submitted is correct.
 - iv. In the **Configure Debugging Data** dialog box, select a table in the left-side table list and click **Auto Sampling**.

- v. In the pop-up dialog box, set the **Data Records** parameter and click **OK**.
 - vi. After the sampling is complete for all tables, click **OK** in the lower-right corner of the **Configure Debugging Data** dialog box.
 - vii. View the debugging data, intermediate result, and debugging result on the **Result** tab.
5. Set the parameters of the stream processing task.
 - i. In the top navigation bar of the configuration tab, click **Task Parameters**.
 - ii. In the Task Parameters pane, set parameters, including the checkpoint mode and interval, based on the sample code of the stream processing task.
 - iii. Click **OK**.
 6. Configure the stream processing task.
 - i. In the top navigation bar of the configuration tab, click **Task Configuration**. The **Task Configuration** pane appears.
 - ii. Set the **Resource Configuration** parameter.
 - If you select **System Recommended**, you do not need to configure resources.
 - If you select **Custom**, you must click **Configure** to configure resources. You can configure resources in one of the following modes: **Visualization Mode** and **Code Editor**.

Mode	Description
Visualization Mode	To configure resources in visualization mode, perform the following steps: <ol style="list-style-type: none"> a. On the resource configuration page, click Visualization Mode. b. Click the  icon in the upper-right corner of a compute node. c. In the dialog box that appears, set the parameters as required. d. Click OK. e. After you set the execution parameters for all compute nodes, click Save in the upper-right corner of the page.
Code Editor	To configure resources in the code editor, perform the following steps: <ol style="list-style-type: none"> a. On the resource configuration page, click Code Editor. b. In the code editor, set the execution parameters for compute nodes as required. Then, click Save in the upper-right corner of the page.

- iii. In the **Task Configuration** pane, click the  icon to turn on **Auto Optimization** and set the **Maximum CUs** and **Expected Maximum Memory** parameters as required. Then, click **OK**.
7. Set the parameters in the stream processing template referenced by the stream processing task.
 - i. In the top navigation bar of the configuration tab, click **Template Parameters**.
 - ii. In the **Template Parameters** pane, set the **Parameter Value** parameter.
 - iii. Click **OK**.
 8. Save and submit the stream processing task.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream processing task.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream processing task.

- iii. (Optional) Publish the stream processing task.
 - If the current project is in Dev mode, publish the stream processing task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the stream processing task after you submit it.

9.9.3.6.4. Modify a stream processing task

Stream processing tasks are used to process data in real time during data development. This topic describes how to modify a stream processing task.

Prerequisites

Stream processing tasks are created. For more information, see [Create a stream processing task of the FLINK_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
6. In the **Stream Processing Tasks** section, move the pointer over the  icon next to the stream processing task that you want to modify and select **Change**.
7. (Optional) Steal the lock of the stream processing task.
 - i. Steal the lock.
 - If the stream processing task is locked by yourself, you do not need to steal the lock.
 - If the stream processing task is locked by another user, click the  icon in the upper-right corner to steal the lock.
 - ii. After the lock is stolen, go to the next step to modify the code, parameters, and configuration of the stream processing task.
8. Modify the code, parameters, and configuration of the stream processing task. For more information, see [Create a stream processing task of the FLINK_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).
9. Save and submit the stream processing task.
 - i. Click the  icon in the upper-right corner of the configuration tab to save the stream processing task.
 - ii. Click the  icon in the upper-right corner of the configuration tab to submit the stream processing task.

- iii. (Optional) Publish the stream processing task.
 - If the current project is in Dev mode, publish the stream processing task to the corresponding project in Prod mode. For more information, see [Publishing management](#).
 - If the current project is in Basic mode, you do not need to publish the stream processing task after you submit it.

9.9.3.6.5. View the version information about a stream processing task

This topic describes how to view the version information about a stream processing task.

Prerequisites

Stream processing tasks are created. For more information, see [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the **Data Processing** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
6. In the **Stream Processing Tasks** section, click the stream processing task for which you want to view the version information.
7. In the top navigation bar of the configuration tab that appears, click **Version Details**. The **Version Details** pane appears.
8. Click the  icon in the **Actions** column of a version to view the comparison between the current version and the reference version.
 - a. In the **Version Comparison** dialog box, click the **Code Comparison** tab to view the comparison between the code in the **Current Version** and **Reference Version** sections.
 - b. To roll back to the reference version, click **Roll Back to Reference Version**.
 - c. In the **Tip** message, click **OK**.
 - a. In the **Version Comparison** dialog box, click the **Task Parameter Comparison** tab to view the comparison between the parameter information in the **Current Version** and **Reference Version** sections.
 - b. To roll back to the reference version, click **Roll Back to Reference Version**.
 - c. In the **Tip** message, click **OK**.
9. Click the  icon in the **Actions** column of a version to view the code of the version.

9.9.3.6.6. Move a stream processing task

Stream processing tasks are used to process data in real time during data development. This topic describes how to move a stream processing task.

Prerequisites

Stream processing tasks are created. For more information, see [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon. The **Stream Processing Tasks** section appears.
6. In the **Stream Processing Tasks** section, move the pointer over the  icon next to the stream processing task that you want to move and select **Move**.
7. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
8. Click **OK**.

9.9.3.6.7. Unpublish or delete a stream processing task

This topic describes how to unpublish, unpublish and delete, and delete stream processing tasks in different states.

Prerequisites

Stream processing tasks are created. For more information, see [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#) or [Create a stream processing task of the FLINK_TEMPLATE_SQL type](#).

Context

- A stream processing task may be in the following states:
 - After you create and save a stream processing task, it enters the **Draft** state.
 - After you submit a stream processing task, it enters the **Submitted** state.
 - After you modify and save a stream processing task in the **Submitted** state, it enters the **Developing** state.
 - After you unpublish a stream processing task in the **Submitted** state, it enters the **Draft** state.
- You can unpublish stream processing tasks only when they are in the **Developing** or **Submitted** state.
- You can delete stream processing tasks only when they are in the **Draft**, **Developing** or **Submitted** state.

Unpublish a stream processing task

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. (Optional) On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode. You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Data Processing** tab in the left-side navigation pane.
5. On the Data Processing tab, move the pointer over the left-side navigation submenu and click the  icon.

The **Stream Processing Tasks** section appears.

6. In the **Stream Processing Tasks** section, move the pointer over the  icon next to the stream processing task that you want to unpublish and select **Unpublish**.
7. In the **Tip** dialog box, enter your comments.
8. Click **OK**.

Delete a stream processing task

1. In the **Stream Processing Tasks** section, move the pointer over the  icon next to the stream processing task that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.9.4. Ad hoc query

9.9.4.1. Overview

Dataphin provides the ad hoc query feature for you to query data. Dataphin can automatically identify and change the SQL syntax based on your computing engine type.

Dataphin uses a business unit as the namespace for its unique logical table models. Therefore, Dataphin uses the following SQL syntax for logical tables:

- Logical tables are referenced in SQL statements in the format of `Business unit.Logical table`.
- The query logic can be in the format of `[Business unit.Logical table.Dimension-associated role. ... Dimension-associated role.Logical dimension table field]`. If the query logic is used as a condition of a **SELECT** statement or a **WHERE** clause, you only need to enter the logic in the format of `[Logical table.Dimension-associated role. ... Dimension-associated role.Logical dimension table field]` to search for fields associated with dimensions.

 **Note** The primary key field of a virtual dimension cannot be used in SQL statements. A virtual dimension can be used only to construct a statistic granularity.

9.9.4.2. Create an ad hoc query task

This topic describes how to create an ad hoc query task.

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode.

You can skip this step if the current project is in **Dev** or **Basic** mode.

4. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
5. Create a folder for storing ad hoc query tasks.
 - i. On the **Ad Hoc Query** tab, click the  icon next to **Ad Hoc Query**.
 - ii. In the **Create Folder** dialog box, enter a folder name and select a folder.
 - iii. Click **OK**.

6. Create an ad hoc query task.
 - i. Open the **Create Item** dialog box by using one of the following methods:
 - In the left-side navigation pane, click the  icon next to the project name and select **Ad Hoc Query**.
 - On the **Ad Hoc Query** tab, click the  icon next to Ad Hoc Query.
 - In the right-side workspace of the **Develop** page, click the  icon next to **SQL Query** in the **Ad Hoc Query** section.
 - ii. In the **Create Item** dialog box, set the **Name** and **Description** parameters and select the created folder.
 - iii. Click **OK**. The Code Editor tab appears.
7. Write the code of the ad hoc query task on the Code Editor tab.
 - i. Write SQL statements to create an ad hoc query task.
 - ii. Click **Precompile** in the upper-right corner of the Code Editor tab to check whether the SQL statements comply with the standard.

If the SQL statements do not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statements so that they comply with the standard.
 - iii. Click **Run** in the upper-right corner of the Code Editor tab to execute the SQL statements. Then, check whether the SQL statements are executed based on the information on the Console tab in the lower part of the page.
8. If the SQL statements are executed, click the  icon in the upper-right corner of the Code Editor tab.

9.9.4.3. Manage ad hoc query tasks

This topic describes how to edit, run, delete, move, and rename an ad hoc query task.

Edit an ad hoc query task

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Dev** or **Basic** mode.

You can skip this step if the current project is in **Dev** or **Basic** mode.
4. On the **Develop** page, click the **Ad Hoc Query** tab in the left-side navigation pane.
5. On the **Ad Hoc Query** tab, move the pointer over the  icon next to the ad hoc query task that you want to edit and select **Change**.
6. (Optional) In the upper-right corner of the configuration tab that appears, click the  icon to steal the lock for the ad hoc query task.

If you need to edit an ad hoc query task that is created by another user or an ad hoc query task that is created by yourself but has been locked by another user, you must steal the lock before you can edit the ad hoc query task.

 - If the ad hoc query task is created by another user or the ad hoc query task is created by yourself but has been locked by another user, you must steal the lock before you can edit the ad hoc query task.

- o If the ad hoc query task is locked by yourself, you do not need to steal the lock.
 - 7. i. In the code editor or on the configuration tab, edit the SQL statements of the ad hoc query task.
In the SQL statements, reference tables based on the following rules:
 - When you query data in a physical table in another project, prefix the table name with the corresponding project name in SQL statements. For example, in the `select * from cloudtest_dev.table4` statement, `cloudtest_dev` indicates the project name and `table4` indicates the name of the physical table.
 - When you query data in a logical table, prefix the table name with the corresponding business unit name in SQL statements. For example, in the `select province from ld_practice.dim_province` statement, `ld_practice` indicates the business unit name and `dim_province` indicates the name of the logical table.
 - When you query data in the development environment, suffix the name of the business unit or project in Prod mode with `_dev`. Dataphin automatically generates the corresponding variable for the business unit or project in Prod mode. For example, if you reference a business unit named `LD_Trade`, Dataphin automatically generates the business unit variable `${LD_Trade}`. By default, this variable is replaced with `LD_Trade_dev` when the SQL statements are executed in the development environment, and replaced with `LD_Trade` when the SQL statements are executed in the production environment. You can assign a specific value for the variable when you write the SQL statements. This variable helps improve the flexibility of the SQL statements when they are executed in different environments.
 - ii. Click **Precompile** in the upper-right corner of the configuration tab to check whether the SQL statements comply with the standard.
If the SQL statements do not comply with the standard, click **Beautify** in the upper-right corner. Then, Dataphin automatically adjusts the SQL statements so that they comply with the standard.
 - iii. Click **Run** in the upper-right corner of the configuration tab to execute the SQL statements. Then, check whether the SQL statements are executed based on the information on the Console tab in the lower part of the page.
8. If the SQL statements are executed, click the  icon in the upper-right corner of the configuration tab.

Delete an ad hoc query task

 **Note** You can delete only ad hoc query tasks that have been locked by yourself.

1. On the **Ad Hoc Query** tab, move the pointer over the  icon next to the ad hoc query task that you want to delete and select **Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

Move an ad hoc query task

 **Note** You can move only ad hoc query tasks that have been locked by yourself.

1. On the **Ad Hoc Query** tab, move the pointer over the  icon next to the ad hoc query task that you want to move and select **Move**.
2. In the **Move File** dialog box, set the **Select Directory** parameter to the destination folder.
3. Click **OK**.

Rename an ad hoc query task

 **Note** You can rename only ad hoc query tasks that have been locked by yourself.

1. On the **Ad Hoc Query** tab, move the pointer over the  icon next to the ad hoc query task that you want to rename and select **Rename**.
2. In the field that appears, enter a new name.
3. Press the Enter key.

9.10. Data distilling

9.10.1. Overview

Based on the data accumulated in data modeling, data distilling can complete data correlation and in-depth data mining and generate code and scheduling tasks to extract value from the data of objects. This module can also identify and connect entities, define tags, and be efficiently applied to various types of business.

Initialization of the data distilling module

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the Develop page, move the pointer over **Develop** in the top navigation bar and select **Distilling**. The **Initialization Configuration** page appears.
4. Select a computing engine from the **Computing Engine** drop-down list and click **Test Connectivity**.
If no computing engine is available, create one.
5. After the computing engine passes the connectivity test, click **Confirm and Initialize**. After the initialization is completed, you can develop data on the Distilling page.

Overview of the data distilling module

The **data distilling** module consists of **Behavior Engine** and **Tag Engine**:

- Behavior Engine allows you to configure detailed behaviors and displays behavior statistics.
- Tag Engine supports visualized tag configuration and automatically generates tag scheduling tasks so that you can manage tag tasks in different development states.

Usage notes

The following table describes the system tables in the data distilling module, which cannot be deleted.

Table	Name	Description
oi_behavior_detail_d	Behavior Engine source data access table	The wide table that extracts source data from Behavior Engine.
oi_behavior_detail_d_sample	Behavior Engine sampling table	The sampling table of the oi_behavior_detail_d table, which is used to display data on the behavior dashboard.
oi_behavior_label_value_d	Behavior Engine tagging result table	The intermediate table for behavior tagging.

Table	Name	Description
oi_behavior_label_value_all	Behavior Engine tagging result table	The accumulating table for behavior tagging.
oi_person_label_value_all	Behavior Engine tagging result table	The table that stores the behavior tagging results.
oi_label_value_rule_d	Tagging rule table	The intermediate table for tagging.
oi_dim_behavior_id	Behavior dimension table	The table that stores behavior definitions synchronized from the source data.
oi_behavior_stat_d	Daily behavior data statistical table	The table that stores the behavior data statistics per day.
oi_behavior_stat_sync	90-day behavior data statistical table	The table that stores the behavior data statistics within 90 days.
oi_behavior_object_all	Annual behavior data object accumulating table	The accumulating table that stores the behavior objects and their attributes.
oi_behavior_property_value	Behavior data attribute table	The object value table that stores behaviors, objects, and object values. You can refer to this table when you create tags.

9.10.2. Behavior Engine

9.10.2.1. Overview

This topic describes the features of the Behavior Engine.

The Behavior Engine consists of the Behavioral Elements, Behavior Rule, and Dashboard submodules.

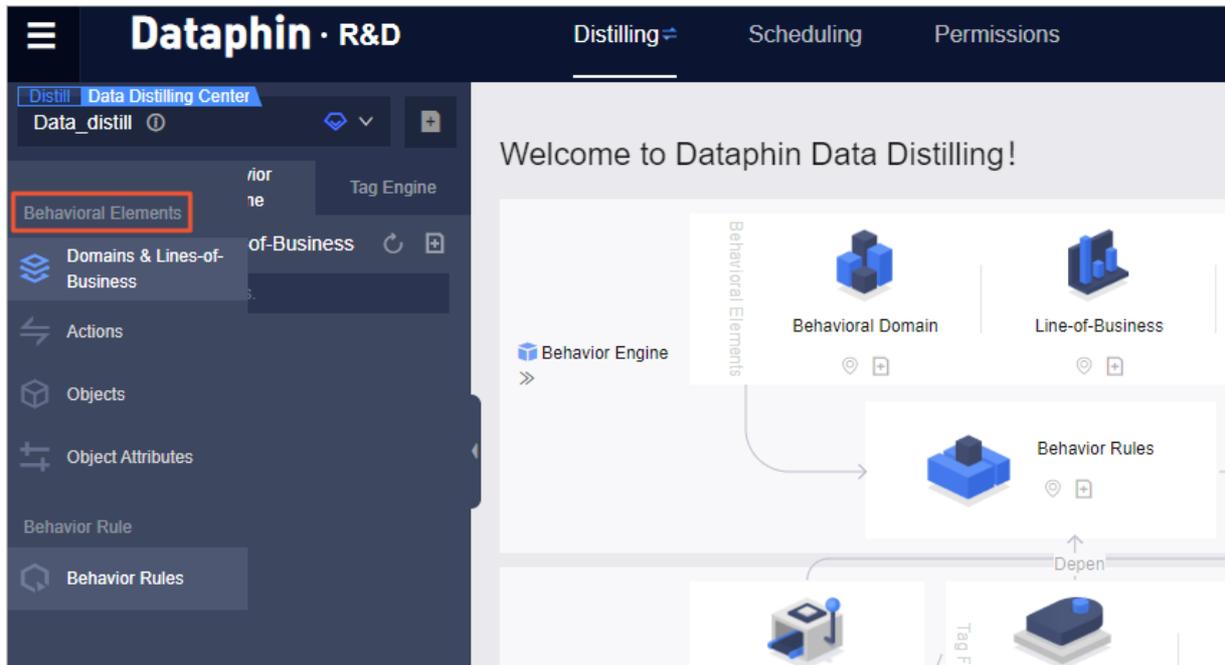
Behavioral Elements submodule

The Behavioral Elements submodule defines standard behavioral elements and their categories. Behavioral elements are categorized into behavioral domains, lines-of-business, actions, objects, and object attributes. You can manage behavioral elements by category.

- Behavioral domain: aggregates behavioral data that has consistent business meanings, such as the e-commerce domain and the entertainment domain.
- Line-of-business: segments behavioral data based on the behavioral domain, such as the Taobao line-of-business and the Tmall line-of-business. Lines-of-business are independent of each other.
- Action: the operation that the behavior subject performs, such as purchasing and browsing.
- Object: the thing on which the behavior subject performs the action, such as a commodity or a movie.
- Object attribute: the description of the object, such as the name, brand, and year.

For example, you can abstract the e-commerce industry as a behavioral domain, and divide the behavioral data in the e-commerce domain into the line-of-business in Mainland China and the line-of-business outside Mainland China by region. The action is payment and the object is commodity. Object attributes include the category, brand, price, size, and color that describe the commodity.

On the Distilling tab, click **Behavior Engine** in the left-side navigation pane. Then, move the pointer over the left-side navigation submenu. You can click a type of behavioral element to manage them on the management page.

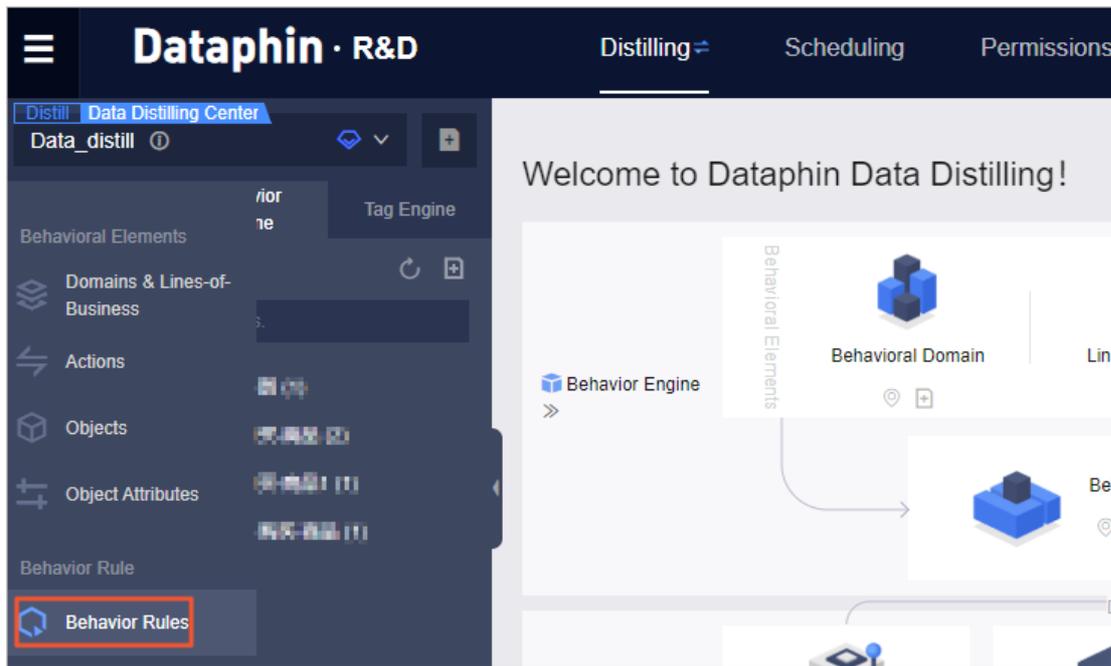


Behavior Rule submodule

Based on the defined behavioral elements, you can configure the corresponding data source and data processing and cleansing rules. Dataphin provides a data source rule for each behavior that is based on a behavioral domain, a line-of-business, an action, and an object to match standardized and structured behaviors with the actual data.

A behavior rule is determined based on the unique behavior and source table. When you preview, save, or submit a behavior rule, Dataphin checks whether the specific behavior and source table are unique.

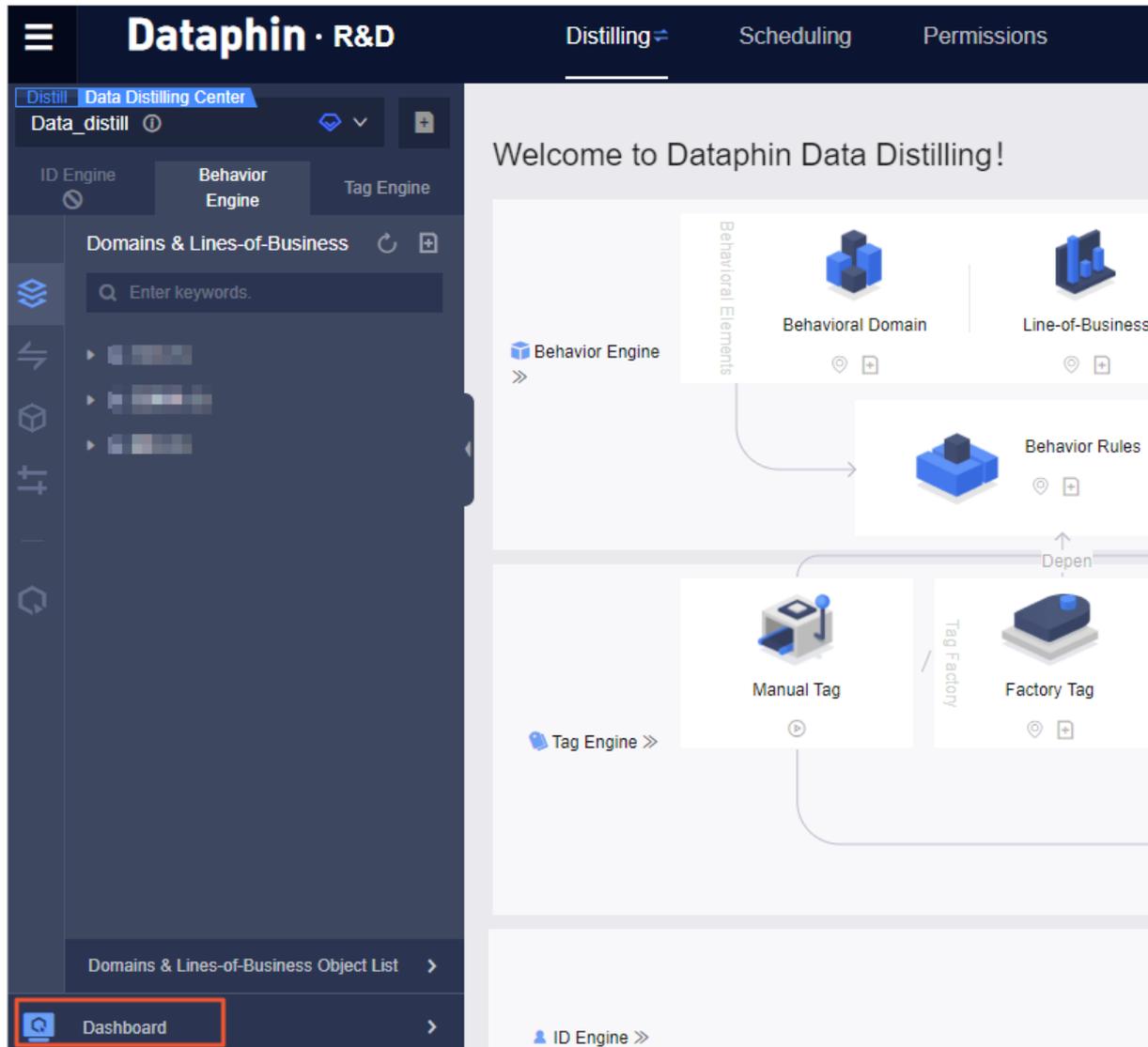
On the Distilling tab, click **Behavior Engine** in the left-side navigation pane. Then, move the pointer over the left-side navigation submenu. You can click **Behavior Rules** to go to the management page.



Dashboard submodule

The Dashboard submodule displays the distribution of behavioral data and the sampling data. It helps you dynamically obtain standardized and structured aggregation results of all behaviors centered on target objects.

On the Distilling tab, click **Behavior Engine** in the left-side navigation pane. Then, click **Dashboard** at the bottom. The Dashboard tab appears.



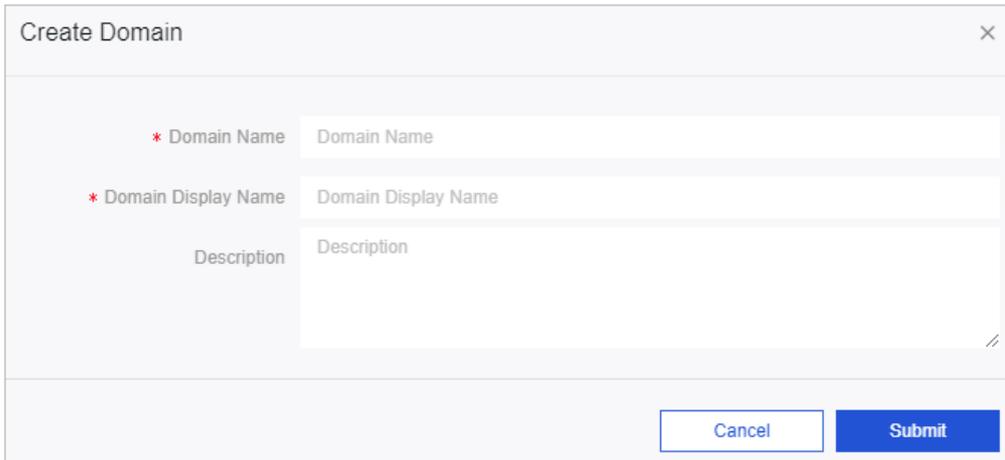
9.10.2.2. Manage behavioral domains

Behavioral domains, such as the e-commerce domain and the entertainment domain, aggregate behavioral data that has consistent business meanings. This topic describes how to create, edit, and delete a behavioral domain.

Create a behavioral domain

1. [Log on to the Dataphin console.](#)
2. Go to the **Domains & Lines-of-Business** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Domains & Lines-of-Business** section appears.
3. Open the **Create Domain** dialog box by using one of the following methods:
 - o In the **Domains & Lines-of-Business** section, click the  icon next to **Domains & Lines-of-Business** and

- select **Behavioral Domain**.
 - o In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Domain**.
 - o In the right-side workspace of the Distilling page, click the  icon under **Behavioral Domain**.
 - o Click **Behavior Rules Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, click **Create Domain** in the upper-right corner.
4. In the **Create Domain** dialog box, set the **Domain Name**, **Domain Display Name**, and **Description** parameters.



The screenshot shows a 'Create Domain' dialog box with a close button (X) in the top right corner. It features three input fields: 'Domain Name' (marked with a red asterisk), 'Domain Display Name' (marked with a red asterisk), and 'Description'. At the bottom right, there are 'Cancel' and 'Submit' buttons.

 **Note** The values of the **Domain Name** and **Domain Display Name** parameters must be unique.

- 5. Click **Submit**.
- 6. In the **Description** dialog box, enter your comments.
- 7. Click **OK**.

Edit a behavioral domain

- 1. Open the **Change Domain** dialog box by using one of the following methods:
 - o In the **Domains & Lines-of-Business** section, click the behavioral domain that you want to edit. On the **View Attributes** tab, click **Change**.
 - o In the **Domains & Lines-of-Business** section, move the pointer over the  icon next to the behavioral domain that you want to edit and select **Change**.
 - o Click **Behavior Rules Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, find the behavioral domain that you want to edit and click the  icon in the **Actions** column.
- 2. In the **Change Domain** dialog box, modify the **Domain Name**, **Domain Display Name**, and **Description** parameters.

3. Click **Submit**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete a behavioral domain

1. Open the **Tip** dialog box for deleting a behavioral domain by using one of the following methods:
 - In the **Domains & Lines-of-Business** section, click the behavioral domain that you want to delete. On the **View Attributes** tab, move the pointer over the  icon and select **Delete**.
 - In the **Domains & Lines-of-Business** section, move the pointer over the  icon next to the behavioral domain that you want to delete and select **Delete**.
 - Click **Behavior Rules Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, find the behavioral domain that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

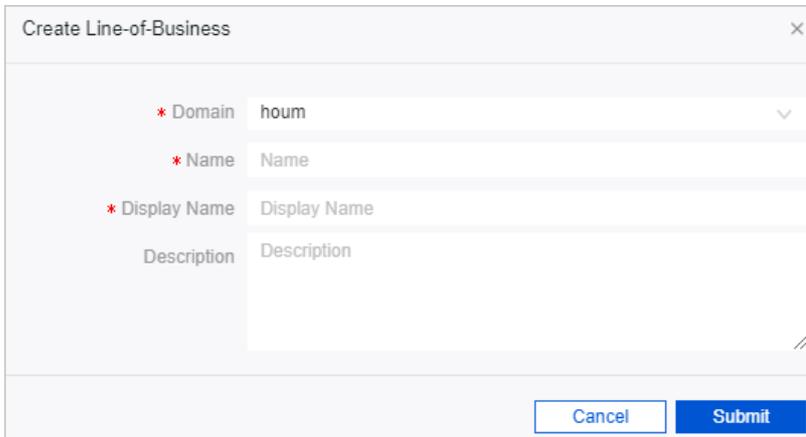
9.10.2.3. Manage lines-of-business

This topic describes how to create, edit, and delete a line-of-business.

Create a line-of-business

1. [Log on to the Dataphin console](#).
2. Go to the **Domains & Lines-of-Business** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Domains & Lines-of-Business** section appears.
3. Open the **Create Line-of-Business** dialog box by using one of the following methods:

- In the **Domains & Lines-of-Business** section, click the  icon next to Domains & Lines-of-Business and select **Line-of-Business**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Line-of-Business**.
 - In the right-side workspace of the Distilling page, click the  icon under **Line-of-Business**.
 - Click **Domains & Lines-of-Business Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, find the behavioral domain for which you want to create a line-of-business and click the  icon in the **Actions** column.
4. In the **Create Line-of-Business** dialog box, set the **Domain**, **Name**, **Display Name**, and **Description** parameters.



 **Note**

- The values of the **Name** and **Display Name** parameters must be unique.
- A line-of-business depends on a behavioral domain. When you create a line-of-business, select the behavioral domain where the line-of-business resides from the drop-down list.

5. Click **Submit**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Edit a line-of-business

1. Open the **Change Line-of-Business** dialog box by using one of the following methods:
- In the **Domains & Lines-of-Business** section, click the line-of-business that you want to edit. On the **View Attributes** tab, click **Change**.
 - In the **Domains & Lines-of-Business** section, move the pointer over the  icon next to the line-of-business that you want to edit and select **Change**.
 - Click **Domains & Lines-of-Business Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, find the line-of-business that you want to edit and click the  icon in the **Actions** column.
2. In the **Change Line-of-Business** dialog box, modify the **Name**, **Display Name**, and **Description** parameters.

3. Click **Submit**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete a line-of-business

Note You cannot delete a line-of-business that is referenced by a behavior rule.

1. Open the **Tip** dialog box for deleting a line-of-business by using one of the following methods:
 - In the **Domains & Lines-of-Business** section, click the line-of-business that you want to delete. On the **View Attributes** tab, move the pointer over the  icon and select **Delete**.
 - In the **Domains & Lines-of-Business** section, move the pointer over the  icon next to the line-of-business that you want to delete and select **Delete**.
 - Click **Domains & Lines-of-Business Object List** in the lower part of the left-side navigation pane. On the **Domains & Lines-of-Business** tab, find the line-of-business that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

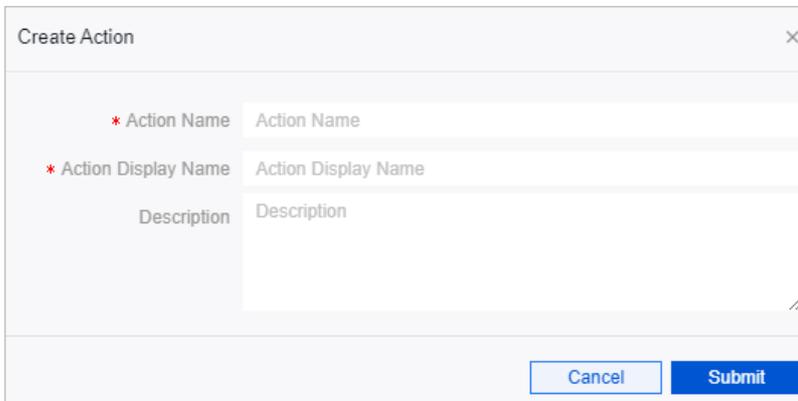
9.10.2.4. Manage actions

Actions, such as purchasing and browsing, are the operations that a behavior subject performs. This topic describes how to create, edit, and delete an action.

Create an action

1. [Log on to the Dataphin console](#).
2. Go to the **Actions** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.

- iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Actions** section appears.
3. Open the **Create Action** dialog box by using one of the following methods:
 - In the **Actions** section, click the  icon next to **Actions**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Actions**.
 - Click **Actions Object List** in the lower part of the left-side navigation pane. On the **Actions** tab, click **Create Action** in the upper-right corner.
 - In the right-side workspace of the **Distilling** page, click the  icon under **Action**.
 4. In the **Create Action** dialog box, set the **Action Name**, **Action Display Name**, and **Description** parameters.



The screenshot shows a 'Create Action' dialog box with a close button (X) in the top right corner. It features three input fields: 'Action Name' (marked with a red asterisk), 'Action Display Name' (marked with a red asterisk), and 'Description'. At the bottom, there are two buttons: 'Cancel' and 'Submit'.

 **Note** The values of the **Action Name** and **Action Display Name** parameters must be unique.

5. Click **Submit**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Edit an action

1. Open the **Change Action** dialog box by using one of the following methods:
 - In the **Actions** section, click the action that you want to edit. On the **View Attributes** tab, click **Change**.
 - In the **Actions** section, move the pointer over the  icon next to the action that you want to edit and select **Change**.
 - Click **Actions Object List** in the lower part of the left-side navigation pane. On the **Actions** tab, find the action that you want to edit and click the  icon in the **Actions** column.
2. In the **Change Action** dialog box, modify the **Action Name**, **Action Display Name**, and **Description** parameters.
3. Click **Submit**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete an action

 **Note** You cannot delete an action that is referenced by a behavior rule.

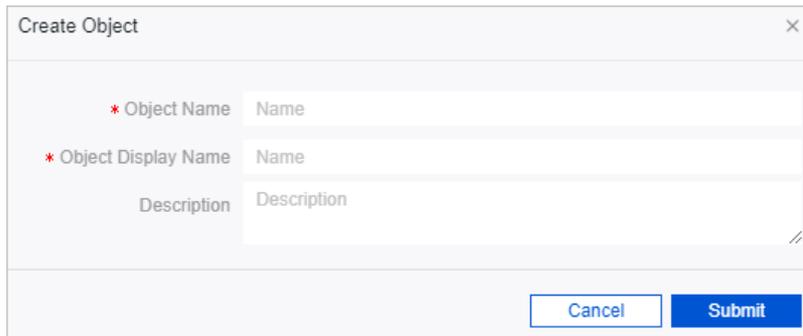
1. Open the **Tip** dialog box for deleting an action by using one of the following methods:
 - In the **Actions** section, click the action that you want to edit. On the **View Attributes** tab, click **Delete**.
 - In the **Actions** section, move the pointer over the  icon next to the action that you want to delete and select **Delete**.
 - Click **Actions Object List** in the lower part of the left-side navigation pane. On the **Actions** tab, find the action that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.10.2.5. Manage objects

Objects, such as a commodity or a movie, are the things on which a behavior subject performs actions. This topic describes how to create, edit, and delete an object.

Create an object

1. [Log on to the Dataphin console.](#)
2. Go to the **Objects** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Objects** section appears.
3. Open the **Create Object** dialog box by using one of the following methods:
 - In the **Objects** section, click the  icon next to **Objects**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Object**.
 - In the right-side workspace of the **Distilling** page, click the  icon under **Object**.
 - Click **Objects Object List** in the lower part of the left-side navigation pane. On the **Objects** tab, click **Create Object** in the upper-right corner.
4. In the **Create Object** dialog box, set the **Object Name**, **Object Display Name**, and **Description** parameters.



 **Note** The values of the **Object Name** and **Object Display Name** parameters must be unique.

5. Click **Submit**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Edit an object

1. Open the **Change Object** dialog box by using one of the following methods:
 - In the **Objects** section, click the object that you want to edit. On the **View Attributes** tab, click **Change**.
 - In the **Objects** section, move the pointer over the  icon next to the object that you want to edit and select **Change**.
 - Click **Objects Object List** in the lower part of the left-side navigation pane. On the **Objects** tab, find the object that you want to edit and click the  icon in the **Actions** column.
2. In the **Change Object** dialog box, modify the **Object Name**, **Object Display Name**, and **Description** parameters.
3. Click **Submit**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete an object

 **Note** You cannot delete an object that is referenced by a behavior rule.

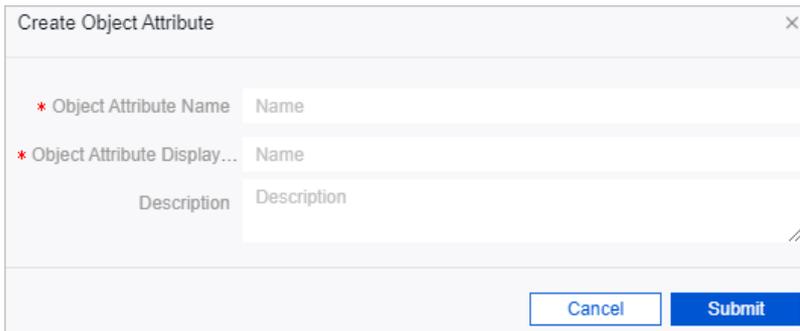
1. Open the **Tip** dialog box for deleting an object by using one of the following methods:
 - In the **Objects** section, click the object that you want to edit. On the **View Attributes** tab, click **Delete**.
 - In the **Objects** section, move the pointer over the  icon next to the object that you want to delete and select **Delete**.
 - Click **Objects Object List** in the lower part of the left-side navigation pane. On the **Objects** tab, find the object that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.10.2.6. Manage object attributes

This topic describes how to create, edit, and delete an object attribute.

Create an object attribute

1. Log on to the Dataphin console.
2. Go to the **Object Attributes** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Object Attributes** section appears.
3. Open the **Create Object Attribute** dialog box by using one of the following methods:
 - In the **Object Attributes** section, click the  icon next to Object Attributes.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Object Attribute**.
 - Click **Object Attributes Object List** in the lower part of the left-side navigation pane. On the **Object Attributes** tab, click **Create Object Attribute** in the upper-right corner.
 - In the right-side workspace of the Distilling page, click the  icon under **Attribute**.
4. In the **Create Object Attribute** dialog box, set the **Object Attribute Name**, **Object Attribute Display Name**, and **Description** parameters.



 **Note** The values of the **Object Attribute Name** and **Object Attribute Display Name** parameters must be unique.

5. Click **Submit**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Edit an object attribute

1. Open the **Change Object Attribute** dialog box by using one of the following methods:
 - In the **Object Attributes** section, click the object attribute that you want to edit. On the **View Attributes** tab, click **Change**.
 - In the **Object Attributes** section, move the pointer over the  icon next to the object attribute that you want to edit and select **Change**.
 - Click **Object Attributes Object List** in the lower part of the left-side navigation pane. On the **Object Attributes** tab, find the object attribute that you want to edit and click the  icon in the **Actions**

column.

2. In the **Change Object Attribute** dialog box, modify the **Object Attribute Name**, **Object Attribute Display Name**, and **Description** parameters.

 **Note** The values of the **Object Attribute Name** and **Object Attribute Display Name** parameters must be unique.

3. Click **Submit**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete an object attribute

 **Note** You cannot delete an object attribute that is referenced by a behavior rule.

1. Open the **Tip** dialog box for deleting an object attribute by using one of the following methods:
 - In the **Object Attributes** section, click the object attribute that you want to delete. On the **View Attributes** tab, click **Delete**.
 - In the **Object Attributes** section, move the pointer over the  icon next to the object attribute that you want to delete and select **Delete**.
 - Click **Object Attributes Object List** in the lower part of the left-side navigation pane. On the **Object Attributes** tab, find the object attribute that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

9.10.2.7. Create a behavior rule

This topic describes how to create and configure a behavior rule.

1. [Log on to the Dataphin console](#).
2. Go to the **Behavior Rules** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, move the pointer over the left-side navigation submenu and click the  icon. The **Behavior Rules** section appears.
3. Create a behavior rule.

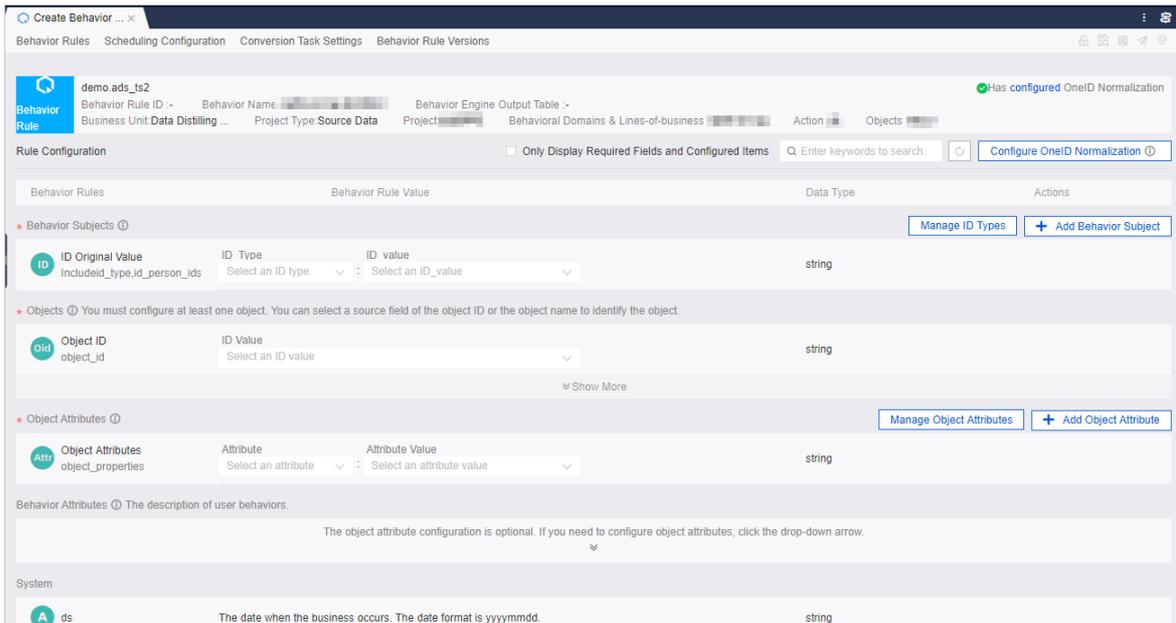
- i. Open the **Create Behavior Rule** dialog box by using one of the following methods:
 - In the **Behavior Rules** section, click the  icon next to Behavior Rules.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Behavior Engine > Behavior Rule**.
 - Click **Behavior Rules Object List** in the lower part of the left-side navigation pane. On the **Behavior Rules** tab, click **Create Behavior Rule** in the upper-right corner.
 - In the right-side workspace of the Distilling page, click the  icon under **Behavior Rules**.
- ii. In the **Create Behavior Rule** dialog box, set the parameters as required.

Section	Parameter	Description
Basic information section	Behavioral Domain and Line-of-Business	Select a behavioral domain and a line-of-business.
	Actions	Select an action.
	Object	Select an object.
	Source Table Type	Select Physical Tables or Logical Tables .

Section	Parameter	Description
Source Table Settings	Main Source Table	<ul style="list-style-type: none"> If you set the Source Table Type parameter to Physical Tables, select the main source table from the drop-down list. If you set the Source Table Type parameter to Logical Tables, select a business unit and then select the main source table from the drop-down lists respectively.
	Filter Conditions	Enter filter conditions, such as <code>ds=\${bizdate}</code> , used to cleanse data records in the source table.

Note If you select a main source table in the development environment, the system cannot automatically parse the task dependencies during scheduling configuration.

- iii. Click **OK** to go to the rule configuration tab.
4. On the rule configuration tab, set the parameters as required.



GUI element	Description

GUI element	Description
<p>Configure OneID Normalization</p>	<ul style="list-style-type: none"> ◦ If you have an ID inverted table, perform the following steps to import the ID inverted table: <ol style="list-style-type: none"> a. Click Configure OneID Normalization. b. In the dialog box that appears, select the ID inverted table. c. Click OK. <div style="background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Note You can select the ID inverted table only in the current project.</p> </div> <ul style="list-style-type: none"> ◦ If you do not have an ID inverted table, go to the Data Processing tab of the Data_distill project on the Develop page to create a task and import the ID inverted table. Make sure that the output name of the task node for the ID inverted table is in the format of Project name.Table name, for example, Data_distill.Inverted table name. <div style="background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Note Only the super administrator can enable or disable ID normalization. After ID normalization is enabled, the result table generates records in one of the following ways:</p> <ul style="list-style-type: none"> ◦ If the selected ID type of a behavior subject is normalized and contained in the ID inverted table, the result table generates records based on OneID and the ID type. ◦ If the ID type of the behavior subject is not normalized or is not contained in the ID inverted table, the result table generates records based on the ID type. </div>
<p>Behavior Subjects</p>	<p>Select an ID type from the ID_Type drop-down list and select an ID value from the ID_value drop-down list.</p> <ul style="list-style-type: none"> ◦ You can click Manage ID Types. In the Manage ID Types dialog box, click Add ID Type to add a custom ID type. The built-in ID types include OneID, Mobile, UserID, Email, and IPAddress. ◦ To add one or more behavior subjects, click Add Behavior Subject. You can delete an existing behavior subject only when more than one behavior subject exists. <div style="background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Note You can delete a custom ID type, but cannot delete or modify the built-in ID types.</p> </div>
<p>Objects</p>	<p>Select an ID value from the ID Value drop-down list and select a display name for the object from the Object Display Name drop-down list.</p> <div style="background-color: #e6f2ff; padding: 5px; margin: 10px 0;"> <p> Note The object ID and object name are used only as object identifiers instead of object attributes. If you need to use the object ID or display name as an object attribute, add the object ID or display name in the Object Attributes section.</p> </div>

GUI element	Description
<p>Object Attributes</p>	<p>An object attribute is the factual description of an object, such as the name, year, or director of a video object. Select an object attribute from the Attribute drop-down list and select an attribute value from the Attribute Value drop-down list.</p> <ul style="list-style-type: none"> ◦ If no object attribute is available, you can click Manage Object Attributes to go to the Object Attributes tab. On the Object Attributes tab, you can create object attributes and manage existing object attributes. For more information, see Manage object attributes. ◦ You can click Add Object Attribute to add one or more object attributes. You can also delete existing object attributes. However, you must retain at least one object attribute. <div style="border: 1px solid #add8e6; padding: 5px; margin-top: 10px;"> <p> Note Make sure that the source fields of the attribute values are stable to prevent the subsequent tag calculation results from being affected by the attribute value changes. Generally, each attribute value of an object is stable and unique. You can select an object attribute that has been defined as a behavioral element from the Attribute drop-down list.</p> </div>
<p>Behavior Attributes</p>	<p>Optional. If the source table you select contains a field that indicates the number of occurrences of the behavior, you can click the  icon and select an attribute value from the Attribute Value drop-down list to add the field as a behavior attribute.</p>
<p>System</p>	<p>The partition field that is automatically created based on the system data timestamp for the Behavior Engine output table, in the format of <code>yyyymmdd</code>.</p>

5. Configure the scheduling policy.
 - i. In the top navigation bar of the rule configuration tab, click **Scheduling Configuration**.
 - ii. In the **Scheduling Configuration** pane, set the parameters as required. The **Node Name**, **Node ID**, **Owner**, and **Description** parameters in the **Basic Information** section cannot be modified.

Scheduling Configuration

Basic Information

Node Name: oi_behavior_detail_d_26

Node ID: #_110281184120881...

Node Type: MaxCompute_BCL

Owner: dataphin@dataphin.com

Description: Scheduling Configuration for Behavior Rule 26

Priority: Medium Priority

Parameter Configuration: Specify values for the parameters that are written in the code of filtering conditions and [Parameters](#)
and [Descriptions](#)

Scheduling Configuration

Schedule Mode: Normal Pause Scheduling

Recurrence: Day 00:00 ⌚

Dependency ⓘ Parsed Dependency

Upstream Dependency + Add Upstream Dependency

Parent Node Outp...	Node Name	Node ID (Instance ID)	Owner	Actions
oi_schedule_root...	oi_schedule_root_n...	#_1107000413816...	dataphin@dataphin.com	🗑️

Current Node

Output Name	Node Name	Node ID	Owner	Actions
oi_behavior_detail...	oi_behavior_detail_...	#_110281184120881...	dataphin@dataphin.com	🔄

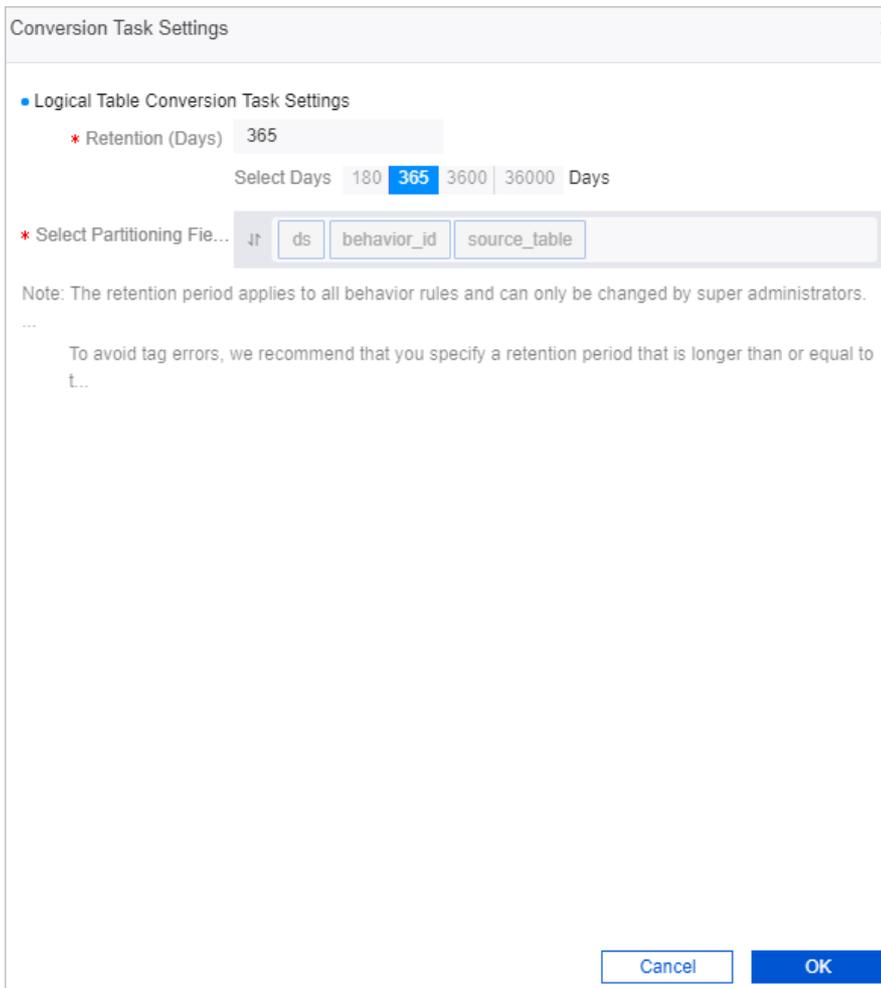
Cancel
OK

Section	Parameter	Description
Basic Information	Priority	<p>Dataphin automatically generates the node name, node ID, node type, owner, and description. You cannot modify these parameters. You can select the priority of the recurring tasks corresponding to the behavior rule. Valid values:</p> <ul style="list-style-type: none"> ▪ Lowest Priority ▪ Low Priority ▪ Medium Priority ▪ High Priority ▪ Highest Priority

Section	Parameter	Description
	Parameter Configuration	You can set variables for the filter conditions of the source table. For more information, see the page that appears after you click Parameters and Descriptions .
Scheduling Configuration	Schedule Mode	You can set the Schedule Mode parameter to Normal or Pause Scheduling .
	Recurrence	<p>You can set the Recurrence parameter to Day, Month, or Week.</p> <ul style="list-style-type: none"> ▪ If you select Day, click the  icon to specify the time of a day. ▪ If you select Month, select a date from the drop-down list and then click the  icon to specify the time of a day. ▪ If you select Week, select a day in a week from the drop-down list and then click the  icon to specify the time of a day.
Dependency	Upstream Dependency	<ul style="list-style-type: none"> ▪ You can click Parsed Dependency. Dataphin automatically parses the upstream dependency based on the selected source table. ▪ You can also add the upstream dependency by performing the following steps: <ul style="list-style-type: none"> a. Click Add Upstream Dependency. b. In the Add Upstream Dependency dialog box, enter a node name or a keyword in the search box to search for the node. c. Select the node and click OK. <div style="background-color: #e1f5fe; padding: 10px; margin-top: 10px;"> <p> Note</p> <ul style="list-style-type: none"> ▪ If the source table is a logical table, the parent node parsed by the system cannot be deleted. ▪ If the source table is a physical table, the parent node parsed by the system can be deleted. ▪ If the automatic parsing fails, you can click Parsed Dependency to parse the dependency again. The parent node that you manually added will not be overwritten by the parsed parent node. </div>
	Current Node	You can click the  icon to view downstream nodes.

- iii. Click **OK**.
- 6. Configure a logical table conversion task.
 - i. In the top navigation bar of the rule configuration tab, click **Conversion Task Settings**.

- ii. In the **Conversion Task Settings** pane, set the **Retention (Days)** and **Select Partitioning Fields** parameters.



- iii. Click **OK**.

7. Save, preview, and submit the behavior rule.

- i. Click the  icon in the upper-right corner of the rule configuration tab to save the configuration of the behavior rule.
- ii. Click the  icon in the upper-right corner of the rule configuration tab to preview the behavior rule.
- iii. Click the  icon in the upper-right corner of the rule configuration tab to submit the behavior rule.

9.10.2.8. Manage behavior rules

This topic describes how to manage behavior rules, for example, how to view, modify, and unpublish a behavior rule as well as generate retroactive data for a behavior rule.

Modify a behavior rule

1. [Log on to the Dataphin console](#).
2. Go to the **Behavior Rules** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.

- iii. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
 - iv. On the **Behavior Engine** tab, click the  icon on the left-side navigation submenu. The **Behavior Rules** section appears.
 3. Go to the **Change Behavior Rules** tab by using one of the following methods:
 - o In the Behavior Rules section, click the behavior rule that you want to modify. On the View Attributes tab, click **Change**.
 - o In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to modify and click the  icon in the **Actions** column.
 - o In the Behavior Rules section, move the pointer over the  icon next to the behavior rule that you want to modify and select **Change**.
 4. On the **Change Behavior Rules** tab, modify the basic information, behavior parameters, scheduling configuration, and conversion task settings of the behavior rule.
 - o Modify the basic information about the behavior rule.
 - a. On the **Change Behavior Rules** tab, click **Behavior Rules** in the top navigation bar.
 - b. In the **Behavior Rules** pane, modify the parameters as required. For more information, see [Create a behavior rule](#).
 - c. Click **OK**.
 - o On the **Change Behavior Rules** tab, modify the behavior parameters as required. For more information, see [Create a behavior rule](#).
 - o Modify the scheduling configuration of the behavior rule.
 - a. On the **Change Behavior Rules** tab, click **Scheduling Configuration** in the top navigation bar.
 - b. In the **Scheduling Configuration** pane, modify the parameters as required. For more information, see [Create a behavior rule](#).
 - c. Click **OK**.
 - o Modify the conversion task settings of the behavior rule.
 - a. On the **Change Behavior Rules** tab, click **Conversion Task Settings** in the top navigation bar.
 - b. In the **Conversion Task Settings** pane, modify the parameters as required. For more information, see [Create a behavior rule](#).
 - c. Click **OK**.
 5. Save, preview, and submit the behavior rule.
 - i. Click the  icon in the upper-right corner to save the configuration of the behavior rule.
 - ii. Click the  icon in the upper-right corner to preview the behavior rule.
 - iii. Click the  icon in the upper-right corner to submit the behavior rule.

View a behavior rule

Go to the **Behavior Rules** section. Then, view the information about a behavior rule by using one of the following methods:

- In the **Behavior Rules** section, click the behavior rule that you want to view. On the View Attributes tab, you can view the information about the behavior rule in the **Basic Information**, **Consumption Information**, **Behavior Subjects**, **Objects**, **Object Attributes**, and **Behavior Attributes** sections.
- In the lower part of left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules**

tab, filter behavior rules by setting the **Status**, **Data Domain**, **Line of Business**, **Action**, **Object**, and **Last Refresh Status** parameters. Then, you can view the parameters such as **Behavior Name**, **Rule ID**, **Rule Source Table**, and **Status** of each behavior rule.

- In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to view and click the ID in the **Rule ID** column. On the **View Attributes** tab, you can view the information about the behavior rule in the **Basic Information**, **Consumption Information**, **Behavior Subjects**, **Objects**, **Object Attributes**, and **Behavior Attributes** sections.

Generate retroactive data for a behavior rule

Note

- You can generate retroactive data for a behavior rule only if the behavior rule is in the **Submitted** or **Developing** state.
- If a retroactive data generation instance is being run for a behavior rule of a behavior, no retroactive data generation instances can be run for other behavior rules of the behavior. That is, only one retroactive data generation instance can be run for a single behavior at a time.

1. Go to the **Behavior Rules** section. Then, open the **Generate Retroactive Data for Behavior Rules** dialog box by using one of the following methods:
 - In the **Behavior Rules** section, move the pointer over the  icon next to the behavior for whose behavior rule you want to generate retroactive data and click **Refresh**.
 - In the **Behavior Rules** section, move the pointer over the  icon next to the behavior rule for which you want to generate retroactive data and select **Refresh**.
 - In the **Behavior Rules** section, click the behavior rule for which you want to generate retroactive data. On the **View Attributes** tab, click **Generate Retroactive Data**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule for which you want to generate retroactive data and click the  icon in the **Actions** column.
2. In the **Generate Retroactive Data for Behavior Rules** dialog box, set parameters in the **Effective Period** and **Instance Name** sections.
3. Click **OK**.

View the behavior rule task under a behavior rule

 **Note** You can view the behavior rule task under a behavior rule only if the behavior rule is in the **Submitted** or **Developing** state.

1. Go to the **Behavior Rules** section. Then, go to the **Behavior Rule Tasks** section on the **Distilling Management** tab by using one of the following methods:
 - In the **Behavior Rules** section, move the pointer over the  icon next to the behavior rule whose behavior rule task you want to view and select **Behavior Rule Tasks**.
 - In the **Behavior Rules** section, click the behavior rule whose behavior rule task you want to view. On the **View Attributes** tab, click the  icon and select **Behavior Rule Tasks**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule whose behavior rule task you want to view and click the  icon in the

Actions column.

2. In the **Behavior Rule Tasks** section, view the details about the behavior rule task under the behavior rule.

Clone a behavior rule

1. Go to the **Behavior Rules** section. Then, go to the Clone Behavior Rule tab by using one of the following methods:
 - In the Behavior Rules section, move the pointer over the  icon next to the behavior rule that you want to clone and select **Clone**.
 - In the Behavior Rules section, click the behavior rule that you want to clone. On the View Attributes tab, click the  icon and select **Clone**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to clone, move the pointer over the  icon in the **Actions** column, and then select **Clone**.
2. On the **Clone Behavior Rule** tab, modify the parameters as required. For more information, see [Modify a behavior rule](#).

Unpublish a behavior rule

Note

- You can unpublish a behavior rule only if the behavior rule is in the **Submitted** or **Developing** state.
- After you unpublish a behavior rule, the system does not automatically delete the partition data of this rule from Behavior Engine output table `oi_behavior_detail_d`. If you do not need to use the data of this rule when you create downstream tags, we recommend that you run code to delete the data. For more information, see [Delete the partition data of a behavior rule](#).

1. Go to the **Behavior Rules** section. Then, open the **Tip** message for confirming the unpublishing of a behavior rule by using one of the following methods:
 - In the Behavior Rules section, move the pointer over the  icon next to the behavior rule that you want to unpublish and select **Unpublish**.
 - In the Behavior Rules section, click the behavior rule that you want to unpublish. On the View Attributes tab, click the  icon and select **Unpublish Behavior Rule**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to unpublish, move the pointer over the  icon in the **Actions** column, and then select **Unpublish**.
2. In the **Tip** message, click **OK**.

Unpublish and delete a behavior rule

Note

- You cannot unpublish and delete a behavior rule on which a tag in the **Submitted** or **Developing** state depends.
- After you unpublish and delete a behavior rule, the system does not automatically delete the partition data of this rule from Behavior Engine output table `oi_behavior_detail_d`. If you do not need to use the data of this rule when you create downstream tags, we recommend that you run code to delete the data. For more information, see [Delete the partition data of a behavior rule](#).

1. Go to the **Behavior Rules** section. Then, open the **Tip** message for confirming the unpublishing and deletion of a behavior rule by using one of the following methods:
 - In the Behavior Rules section, move the pointer over the  icon next to the behavior rule that you want to unpublish and delete and select **Unpublish and Delete**.
 - In the Behavior Rules section, click the behavior rule that you want to unpublish and delete. On the View Attributes tab, click the  icon and select **Unpublish and Delete**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to unpublish and delete, move the pointer over the  icon in the **Actions** column, and then select **Unpublish and Delete**.
2. In the **Tip** message, click **OK**.

Delete a behavior rule

Note You can delete a behavior rule only if the behavior rule is in the **Draft** state.

1. Go to the **Behavior Rules** section. Then, open the **Tip** message for confirming the deletion of a behavior rule by using one of the following methods:
 - In the Behavior Rules section, move the pointer over the  icon next to the behavior rule that you want to delete and select **Delete**.
 - In the Behavior Rules section, click the behavior rule that you want to delete. On the View Attributes tab, click the  icon and select **Delete**.
 - In the lower part of the left-side navigation pane, click **Behavior Rules Object List**. On the **Behavior Rules** tab, find the behavior rule that you want to delete, move the pointer over the  icon in the **Actions** column, and then select **Delete**.
2. In the **Tip** message, click **OK**.

Delete the partition data of a behavior rule

This section describes how to delete the partition data of a behavior rule.

In the `Data_distill` project, the physical table `oi_behavior_detail_d` has the partitions `ds`, `source_table`, and `behavior_id`.

1. Run the following command to view the partitions of the physical table:

```
show partitions oi_behavior_detail_d;
```

The following output or similar output appears:

```
ds=20191020/source_table=data_distill.data_distill_movie_info/behavior_id=20
ds=20191021/source_table=data_distill.data_distill_movie_info/behavior_id=20
```

2. Run the following command to delete all partitions related to this rule:

```
ALTER TABLE oi_behavior_detail_d
DROP IF EXISTS PARTITION(ds = 'ds', source_table =
'source_table', behavior_id = 'behavior_id');
```

3. Run the following command to check whether any partition is left. If some partitions are left, repeat the preceding two steps until all partitions are deleted.

```
show partitions oi_behavior_detail_d;
```

9.10.2.9. View behavioral data

This topic describes how to view the distribution of behavioral data and the sampling result of behavioral data on the dashboard.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Distilling** page.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
3. On the **Distilling** page, click the **Behavior Engine** tab in the left-side navigation pane.
4. On the **Behavior Engine** tab, click **Dashboard** in the lower part of the left-side navigation pane.
5. On the **Behavior Dashboard** page, view behavioral data.
 - The left-side pane of the **Behavior Dashboard** page displays behavioral domains, lines-of-business, and objects from left to right.
 - Move the pointer over a bar below **Domain** to view the number of behaviors in the corresponding behavioral domain and its percentage in the total number of behaviors in all behavioral domains.
 - Move the pointer over a bar below **Line-of-Business** to view the number of behaviors in the corresponding line-of-business and its percentage in the total number of behaviors in all lines-of-business.
 - Move the pointer over a bar below **Objects** to view the number of behaviors on the corresponding object and its percentage in the total number of behaviors on all objects.
 - The right-side pane of the **Behavior Dashboard** page displays the sampling result of behavioral data. By default, Dataphin displays the sampling result of the behavioral data of yesterday. You can click the  icon next to **Sampling Duration** and select a sampling data timestamp.

9.10.3. Tag Engine

9.10.3.1. Create a factory tag

This topic describes how to create and configure a factory tag.

Prerequisites

A behavior rule is created. For more information, see [Create a behavior rule](#).

Create a factory tag

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **R&D** in the top navigation bar.
3. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
4. On the Distilling page, click the **Tag Engine** tab in the left-side navigation pane. The **Factory Tags** section appears.
5. Open the **Create Factory Tag** dialog box by using one of the following methods:
 - In the Factory Tags section, click the  icon next to **Factory Tags**.
 - In the left-side navigation pane, click the  icon next to the project name and choose **Tag Engine > Factory Tags**.
 - On the right-side workspace, click the  icon below **Factory Tag**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, click **Create Tag** in the upper-right corner.
6. In the **Create Factory Tag** dialog box, set the parameters as required.

Parameter	Description
Tag Name	The name of the factory tag, for example, <code>item_price_favor</code> .
Tag Display Name	The display name of the factory tag, for example, Price preferences in last 30 days .
Description	The description of the factory tag.
Category	The category of the factory tag. If no category is available, you can create a category. For more information, see Manage tag categories .
Publishing Status	Specifies whether to show the factory tag. Valid values: Show and Hide .
Result Table	The name of the result table of the factory tag, for example, <code>item_price_favor</code> .

7. Click **OK**.

Configure the data source and tag value

1. On the **Create Tag** tab, click **Configure** below **Configure Data Source and Tag Value**.
2. In the **Configure Data Source** step, set the parameters as required.

Parameter	Description
Filter by Behavioral Domain	The behavioral domains to which the data sources belong. In this example, select E-commerce .
Filter by Action	The actions that serve as the data sources. In this example, select Purchase-Commodity .
Filter by Common Object Attribute	The shared object attributes of the selected actions. If the selected actions has shared object attributes, you can filter source data based on the shared object attributes. You can select a maximum of three shared object attributes.

Parameter	Description
Data Source - Filter by Time Range	The time range within which the behavioral data is used to produce the factory tag. Valid values: 30 Days , 60 Days , 90 Days , 180 Days , and 365 Days .

3. Click **Next**.

4. In the **Configure Tag Value** step, set the parameters as required.

Parameter	Description
Tagging Method	<p>The method to tag the behavioral data. Valid values:</p> <ul style="list-style-type: none"> ◦ Direct Tagging: Tag the filtered behavioral data based on the tag value you specify. If you select Direct Tagging, you must enter a tag value in the Tag Value field in the Tag Value Configuration - Tag Value Definition section. ◦ Object Attribute Mapping: Use attribute values as tag values or configure rules to map attribute values to tag values. This way, you can add different tags to behavioral data with different attributes at a time. If you select Object Attribute Mapping, you must select options in the Select Common Object Attributes and Select Mapping Method steps.
Shared Object Attributes	The shared object attributes of the selected actions. If you set Tagging Method to Object Attribute Mapping , you must set this parameter. In this example, select Commodity price .
Select Mapping Method	<p>The mapping method between attribute values and tag values. If you set Tagging Method to Object Attribute Mapping, you must set this parameter. Valid values:</p> <ul style="list-style-type: none"> ◦ Apply as Tag Value: Use attributes values as tag values. For example, you can use the attribute value Commodity category as a tag value. ◦ Customize Mapping Rules: Configure rules to map attribute values to tag values. For example, you can configure the following mapping rules: <ul style="list-style-type: none"> ■ Map a value of the Commodity price attribute that is Less Than 200 to the Low price tag value. ■ Map a value of the Commodity price attribute that is Less Than 500 to the Medium price tag value. ■ Map a value of the Commodity price attribute that meets the Other condition to the High price tag value. <p>If multiple mapping rules are applied to the same behavior, an attribute value goes down the list of rules until it hits a rule that applies. Based on the preceding rules, Dataphin uses the following mapping between attribute values and tag values:</p> <ul style="list-style-type: none"> ■ Commodity price<200 : The behavioral data is tagged as Low price. ■ 200≤Commodity price<500 : The behavioral data is tagged as Medium price. ■ 500≤Commodity price : The behavioral data is tagged as High price.

5. Click **Configure Rule for Tag Generation**.

6. In the **Configure Rule for Tag Generation** step, set the parameters as required.

GUI element	Description
Number of Returned Tag Values (Maximum)	The maximum number of tag values that can be returned for each behavior subject ID. Multiple behavioral data records may be generated for each ID. Therefore, each ID may have multiple tag values. Dataphin preferentially returns tag values with higher preferences based on the number of tag values you specify. You can view the tag value preferences in the tag result table.
Select Decay Period	The period of decay. The behaviors that are performed in different periods have different impacts on tag values. The behaviors that are performed in the same period have the same impact on tag values. The impact of behaviors in different periods decreases with the time based on the specified decay period and curve. If the tag values rapidly change with the behavior, such as the diaper model preference, we recommend that you select a shorter period. If the tag values are stable and change slowly, such as individual skin type prediction, we recommend that you select a longer period.
Select Decay Curve	The curve of the decay. You can set the decay mode for the behavioral data within the selected time range, such as 90 days, based on the number of decay periods in the time range. For example, if the decay period lasts seven days, the number of decay periods is 13 ($90/7 \approx 13$). The impact of tag values of behavioral data in different decay periods is related to the decay mode you select. For example, the individual skin is stable for a long period. Therefore, no impact is exerted on the skincare product preference for a consumer who adds skincare products to favorites seven days ago and within the last seven days. In this case, you can select No Decay.
Tag Generation Rules - Assign Weights	The weights of behaviors based on the business judgment. Each weight can have a maximum of two decimal places, and the sum of weights is 1. A larger weight indicates a greater impact of this behavior on the final tag value.

GUI element	Description
Tag Generation Rules - Set Coverage Scope	<p>The scope of covered customers. The following content describes the coverage percentage and coverage scope of customers.</p> <ul style="list-style-type: none"> ○ The coverage percentage of customers is equivalent to the number of customers who perform a behavior less than the specified number of times divided by the total number of customers. For example, the coverage percentage of customers who purchase moisturizing skincare products less than 3 times is 10%. ○ The coverage scope limits the scope of customers whose behavioral data is to be tagged. Only the behavioral data of customers in the coverage scope can be tagged. This helps reduce the impact of occasional behaviors and improve the accuracy of tagging. <ul style="list-style-type: none"> ■ If you do not set the coverage scope, the behavioral data of all customers is processed. ■ If you set a specific coverage scope, the behavioral data of customers who exceed the coverage scope is filtered. <p>Assume that the behavior is purchasing moisturizing skincare products. Among all the customers, 10% of them purchase moisturizing skincare products less than 3 times and 90% of them purchase moisturizing skincare products less than 100 times. You can set the coverage scope to the range from 10% to 90%. As a result, only the behavioral data of customers who purchase moisturizing skincare products more than 3 times and less than 100 times can be tagged (3 and 100 times included). This configuration helps reduce the impact on the accuracy of tagging due to too few or excessive purchases.</p>

7. Click **OK**.

Configure conversion task settings

1. On the Create Tag tab, click **Conversion Task Settings** in the top navigation bar.
2. In the **Conversion Task Settings** pane, set the parameters as required.

Parameter	Description
Retention (Days)	The retention period of the factory tag. You can enter a value in the field or select one of the following options: 90, 180, 365, and 3650 .
Select Partitioning Fields	The partition fields of the result table.

3. Click **OK**.
4. Save, test, and submit the factory tag.
 - i. Click the  icon in the upper-right corner to save the configuration of the factory tag.
 - ii. Click the  icon in the upper-right corner to test the factory tag.
 - iii. Click the  icon in the upper-right corner to submit the factory tag. A recurring task scheduled by day is automatically generated after the factory tag is submitted.

 **Note** After the factory tag is submitted, you must generate retroactive data for the tag to obtain the historical behavioral data within the selected time range as the source data. If you do not generate retroactive data, the result table of the factory tag may be empty.

9.10.3.2. Manage factory tags

This topic describes how to manage factory tags, for example, how to view, modify, unpublish, and unpublish and delete a factory tag as well as generate retroactive data for a factory tag.

View a factory tag

1. [Log on to the Dataphin console.](#)
2. Go to the **Factory Tags** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Tag Engine** tab in the left-side navigation pane. The **Factory Tags** section appears by default.
3. View the information about a factory tag by using one of the following methods:
 - Click the factory tag that you want to view in the **Factory Tags** section. The **View Attributes** tab appears, which provides entries for you to view the following tag information:
 - **Tag version:** Click **Change** on the **View Attributes** tab. On the **Change Tag** tab, click **Tag Version** in the top navigation bar. In the **Tag Version** pane, you can view the versions of the tag.
 - **Conversion task settings:** Click **Change** on the **View Attributes** tab. On the **Change Tag** tab, click **Conversion Task Settings** in the top navigation bar. In the **Conversion Task Settings** pane, you can view the conversion task settings of the tag.
 - **Basic tag information:** Click **Change** on the **View Attributes** tab. On the **Change Tag** tab, click **Tag Information** in the top navigation bar. In the **Change Basic Information** pane, you can view the basic information about the tag.
 - **Result table:** Click the name of the result table in the **Result Table** section on the **View Attributes** tab. On the page that appears, you can view the details about the result table of the tag.
 - **Behavior rule:** Click a behavior rule ID in the **Dependency** section on the **View Attributes** tab. In the pane that appears, you can view the details about the behavior rule on which the tag depends.
 - **Behavior:** Click the behavior ID in the **Dependency** section on the **View Attributes** tab. On the **Behavior Rules** tab, you can view the basic information about the behavior.
 - Click **Factory Tags Object List** in the lower part of the left-side navigation pane. On the **Factory Tags** tab, you can filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters. You can also enter a tag name or tag ID in the search box to search for your desired factory tags.

Modify a factory tag

 **Note** You cannot change the result table name of a factory tag if the tag is in the **Submitted** or **Developing** state.

1. Go to the **Factory Tags** section. Then, go to the **Change Tag** tab by using one of the following methods:
 - In the **Factory Tags** section, move the pointer over the  icon next to the factory tag that you want to modify and select **Change**.
 - In the **Factory Tags** section, click the factory tag that you want to modify. On the **View Attributes** tab, click **Change**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory**

- Tags** tab, find the factory tag that you want to modify and click the  icon in the **Actions** column.
2. On the **Change Tag** tab, modify the basic information about and conversion task settings of the tag.
 - o Modify the conversion task settings of the tag.
 - a. Click **Conversion Task Settings** in the top navigation bar.
 - b. In the **Conversion Task Settings** pane, modify the parameters as required. For more information, see [Create a factory tag](#).
 - c. Click **OK**.
 - o Click **Tag Information** in the top navigation bar. In the **Change Basic Information** pane, modify the parameters as required. For more information, see [Create a factory tag](#).

Generate retroactive data for a factory tag

 **Note** You can generate retroactive data for a factory tag only if the tag is in the **Submitted** or **Developing** state. System

1. Go to the **Factory Tags** section. Then, open the **Generate Retroactive Data for Tags** dialog box by using one of the following methods:
 - o In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, find the factory tag for which you want to generate retroactive data and click the  icon in the **Actions** column.
 - o In the **Factory Tags** section, click the factory tag for which you want to generate retroactive data. On the **View Attributes** tab, click **Refresh**.
2. In the **Generate Retroactive Data for Tags** dialog box, set parameters in the **Effective Period** and **Instance Name** sections.
3. Click **OK**.

View the tag task under a factory tag

 **Note** You can view the tag task under a factory tag only if the tag is in the **Submitted** or **Developing** state.

1. Go to the **Factory Tags** section. Then, go to the **Tag Tasks** section on the **Distilling Management** tab by using one of the following methods:
 - o In the **Factory Tags** section, move the pointer over the  icon next to the factory tag whose tag task you want to view and select **Tag Tasks**.
 - o In the **Factory Tags** section, click the factory tag whose tag task you want to view. On the **View Attributes** tab, move the pointer over the  icon and select **Tag Task**.
 - o In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, find the factory tag whose tag task you want to view and click the  icon in the **Actions** column.
2. In the **Tag Tasks** section on the **Distilling Management** tab, view the details about the tag task under the factory tag.

View the report of a factory tag

 **Note** You can view the report of a factory tag only if the tag is in the **Submitted** or **Developing** state.

1. Go to the **Factory Tags** section. Then, go to the Tag Report pane by using one of the following methods:
 - In the Factory Tags section, move the pointer over the  icon next to the factory tag whose report you want to view and select **Tag Report**.
 - In the Factory Tags section, click the factory tag whose report you want to view. On the View Attributes tab, click **View Tag Report**.
2. In the Tag Report pane, view the tag report.

Clone a factory tag

1. Go to the **Factory Tags** section. Then, go to the **Clone Tag** tab by using one of the following methods:
 - In the Factory Tags section, move the pointer over the  icon next to the factory tag that you want to clone and select **Clone**.
 - In the Factory Tags section, click the factory tag that you want to clone. On the View Attributes tab, move the pointer over the  icon and select **Clone**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab:
 - If the factory tag that you want to clone is in the **Submitted** or **Developing** state, move the pointer over the  icon in the **Actions** column and select **Clone**.
 - If the factory tag that you want to clone is in the **Draft** state, click the  icon in the Actions column.
2. On the **Clone Tag** tab, modify the parameters as required. For more information, see [Modify a factory tag](#).

Unpublish a factory tag

 **Note** You can unpublish a factory tag only if the tag is in the **Submitted** or **Developing** state.

1. Go to the **Factory Tags** section. Then, open the **Tip** dialog box for confirming the unpublishing of a factory tag by using one of the following methods:
 - In the Factory Tags section, move the pointer over the  icon next to the factory tag that you want to unpublish and select **Unpublish**.
 - In the Factory Tags section, click the factory tag that you want to unpublish. On the View Attributes tab, move the pointer over the  icon and select **Unpublish**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, find the factory tag that you want to unpublish, move the pointer over the  icon in the **Actions** column, and then select **Unpublish**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**. After the tag is unpublished, it enters the **Draft** state.

Unpublish and delete a factory tag

 **Note** You can unpublish and delete a factory tag only if the tag is in the **Submitted** or **Developing** state.

1. Go to the **Factory Tags** section. Then, open the **Tip** dialog box for confirming the unpublishing and deletion of a factory tag by using one of the following methods:
 - In the **Factory Tags** section, move the pointer over the  icon next to the factory tag that you want to unpublish and delete and select **Unpublish and Delete**.
 - In the **Factory Tags** section, click the factory tag that you want to unpublish and delete. On the **View Attributes** tab, move the pointer over the  icon and select **Unpublish and Delete**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, find the factory tag that you want to unpublish and delete, move the pointer over the  icon in the **Actions** column, and then select **Unpublish and Delete**.
2. In the **Tip** dialog box, enter your comments.
3. Click **OK**.

Delete a factory tag

 **Note** You can delete a factory tag only if the tag is in the **Draft** state.

1. Go to the **Factory Tags** section. Then, open the **Tip** or **Delete Tag** message.
 - In the **Factory Tags** section, move the pointer over the  icon next to the factory tag that you want to delete and select **Delete**.
 - In the **Factory Tags** section, click the factory tag that you want to delete. On the **View Attributes** tab, click the  icon and select **Delete**.
 - In the lower part of the left-side navigation pane, click **Factory Tags Object List**. On the **Factory Tags** tab, find the factory tag that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** or **Delete Tag** message, click **OK**.

9.10.3.3. Manage the logical tag table

This topic describes how to manage the logical tag table, for example, how to view the logical tag table and modify the factory tags and manual tags in the logical tag table.

Dataphin automatically creates a logical tag table when a distilling project is initialized. The logical tag table collects all the factory tags that are in the **Submitted** state and manual tags in physical tables by `id_type` and `id_value`. Note that manual tags in physical tables are not developed in the tag factory. The name of the logical tag table is fixed to `Ads_distill_labels` and cannot be changed.

View the logical tag table

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Tag Tables** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Tag Engine** tab in the left-side navigation pane.

- iv. On the **Tag Engine** tab, click the  icon on the left-side navigation submenu. The **Logical Tag Tables** section appears.
3. In the **Logical Tag Tables** section, click **Ads_distill_labels**. The **Logical Tag Table** tab appears.

The **Logical Tag Table** tab consists of the **Factory Tags** and **Manual Tags** tabs. All the factory tags in the **Submitted** or **Developing** state are automatically attached to the logical tag table. You can also import manual tags that are developed by coding from physical tables to the logical tag table.

Modify a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag that you want to modify by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Click the  icon in the **Actions** column.
4. On the **Change Tag** tab, modify the parameters as required. For more information, see [Edit a factory tag](#).

Generate retroactive data for a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag for which you want to generate retroactive data by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Click the  icon in the **Actions** column.
4. In the **Generate Retroactive Data for Tags** dialog box, set the parameters as required. For more information, see [Generate retroactive data](#).

View the tag task under a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag whose tag task you want to view by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Click the  icon in the **Actions** column.
4. In the **Tag Tasks** section on the **Distilling Management** tab, view the details about the tag task under the factory tag.

Clone a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag that you want to clone by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Move the pointer over the  icon in the **Actions** column and select **Clone**.
4. On the **Clone Tag** tab, modify the parameters as required. For more information, see [Edit a factory tag](#).

Unpublish a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag that you want to unpublish by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Move the pointer over the  icon in the **Actions** column and select **Unpublish**.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Unpublish and delete a factory tag

1. On the **Logical Tag Table** tab, click **Factory Tags**.
2. On the **Factory Tags** tab, find the factory tag that you want to unpublish and delete by using one of the following methods:
 - Click the  icon. Then, filter factory tags by setting the **Category**, **Development Status**, **Test Run Status**, and **Last Refresh Status** parameters.
 - Enter a tag display name, tag name, or tag ID in the search box to search for your desired factory tag.
 - Find your desired factory tag in the tag list.
3. Move the pointer over the  icon in the **Actions** column and select **Unpublish and Delete**.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

Create a manual tag

1. On the **Logical Tag Table** tab, click **Manual Tags**.
2. On the **Manual Tags** tab, click **Create Manual Tags** in the upper-right corner.
3. In the **Create Manual Tag** dialog box, perform the following steps:

- i. In the **Association Logic** step, set the **Source Physical Table** parameter and parameters in the **Association Logic** section.
 - ii. Click **OK**.
 - iii. In the **Add Fields** step, select the fields that are used to associate the source physical table with the logical tag table and click **Add >>**.
 - iv. On the **New List** tab, set the **Tag Display Name** parameter.
You can also modify the **Tag Name**, **Category**, and **Publishing Status** parameters. To delete a field from the **New List** tab, click the  icon in the **Actions** column. In the message that appears, click **OK**.
 - v. Click **OK**.
4. In the **Description** dialog box, enter your comments and click **OK**.

Modify a manual tag

1. On the **Logical Tag Table** tab, click **Manual Tags**.
2. On the **Manual Tags** tab, find the manual tag that you want to modify by using one of the following methods:
 - Filter manual tags by setting the **Category** parameter.
 - Enter a keyword in the search box to search for manual tags whose names contain the keyword.
 - Find your desired manual tag in the tag list.
3. Click the  icon in the **Actions** column.
4. In the **Modify Manual Tag** dialog box, set the parameters as required.

Parameter	Description
Tag Name	The name of the manual tag. You cannot change the name.
Tag Display Name	The display name of the manual tag. You cannot change the display name.
Description	The description of the manual tag.
Category	The category of the manual tag.
Publishing Status	Specifies whether to show the manual tag.

5. Click **Submit**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Delete a manual tag

1. On the **Logical Tag Table** tab, click **Manual Tags**.
2. On the **Manual Tags** tab, find the manual tag that you want to delete by using one of the following methods:
 - Filter manual tags by setting the **Category** parameter.
 - Enter a keyword in the search box to search for manual tags whose names contain the keyword.
 - Find your desired manual tag in the tag list.

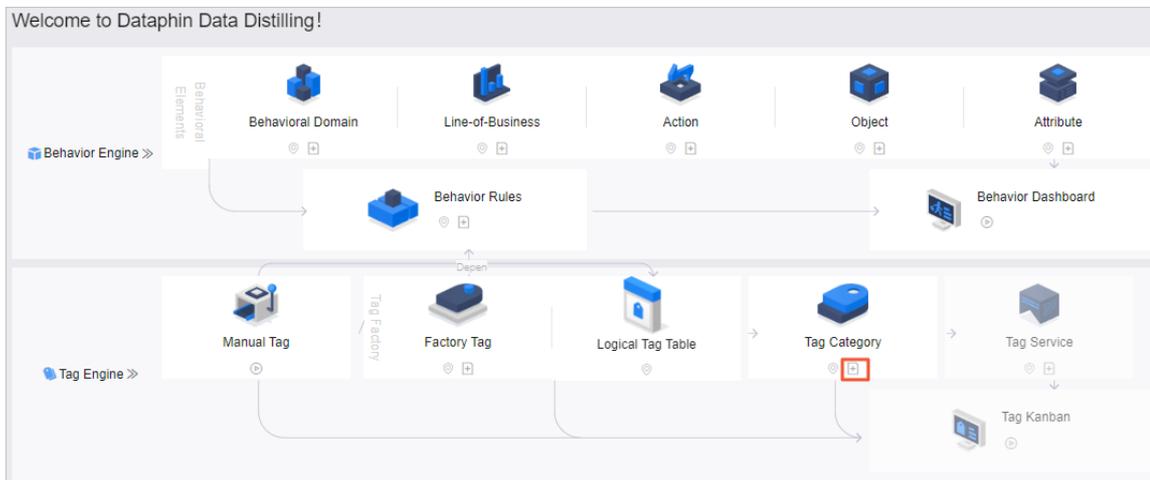
3. Click the  icon in the **Actions** column.
4. In the **Tip** dialog box, enter your comments.
5. Click **OK**.

9.10.3.4. Manage tag categories

This topic describes how to create, modify, and delete a tag category and perform operations on the details page of a tag category.

Create a tag category

1. [Log on to the Dataphin console](#).
2. Go to the **Tag Categories** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - iii. On the **Distilling** page, click the **Tag Engine** tab in the left-side navigation pane.
 - iv. On the **Tag Engine** tab, click the  icon on the left-side navigation submenu. The **Tag Categories** section appears.
3. Open the **Create Category** dialog box by using one of the following methods:
 - o In the **Tag Categories** section, click the  icon next to **Tag Categories**.
 - o In the left-side navigation pane, click the  icon next to the project name and choose **Tag Engine > Tag Categories**.
 - o In the right-side workspace of the **Distilling** page, click the  icon below **Tag Category**.



- o In the lower part of the left-side navigation pane, click **Tag Categories Object List**. On the **Tag Categories** tab, click **Create Category** in the upper-right corner.
4. In the **Create Category** dialog box, set the parameters as required.

Parameter	Description
Display Name	The display name of the tag category. The display name can contain letters, digits, underscores (_), and hyphens (-). However, the display name cannot be a numerical string.
Name	The name of the tag category. The name can contain letters, digits, underscores (_), and hyphens (-). However, the name cannot be a numerical string.
Description	The description of the tag category.

5. Click **OK**.
6. In the **Description** dialog box, enter your comments.
7. Click **OK**.

Modify a tag category

1. Go to the **Tag Categories** section. Then, open the **Modify Category** dialog box by using one of the following methods:
 - In the **Tag Categories** section, move the pointer over the  icon next to the tag category that you want to modify and select **Change**.
 - In the lower part of the left-side navigation pane, click **Tag Categories Object List**. On the **Tag Categories** tab, find the tag category that you want to modify and click the  icon in the **Actions** column.
2. In the **Modify Category** dialog box, modify the parameters as required. For more information, see [Create a tag category](#).
3. Click **OK**.
4. In the **Description** dialog box, enter your comments.
5. Click **OK**.

Delete a tag category

1. Go to the **Tag Categories** section. Then, open the **Tip** dialog box for confirming the deletion of a tag category by using one of the following methods:
 - In the **Tag Categories** section, move the pointer over the  icon next to the tag category that you want to delete and select **Delete**.
 - In the lower part of the left-side navigation pane, click **Tag Categories Object List**. On the **Tag**

- Categories tab, find the tag category that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** dialog box, enter your comments.
 3. Click **OK**.

Perform operations on the details page of a tag category

1. Go to the **Tag Categories** section. Then, click **Tag Categories Object List** in the lower part of the left-side navigation pane. The **Tag Categories** tab appears.
2. Click the name of a tag category in the **Name** column. The details page of the tag category appears, where you can perform the following operations:
 - Click the **Factory Tags** tab and manage factory tags. For example, modify a factory tag, unpublish and delete a factory tag, and generate retroactive data for a factory tag. For more information, see [Manage factory tags](#).
 - Click the **Manual Tags** tab and manage manual tags. For example, modify and delete a manual tag. For more information, see [Modify a manual tag](#) and [Delete a manual tag](#).
 - Click the **Factory Tags** or **Manual Tags** tab, select multiple tags, and then click **Hide All** in the lower part of the tab. In the **Tag Hide** message, click **OK**.
 - Click the **Factory Tags** or **Manual Tags** tab, select multiple tags, and then click **Show All** in the lower part of the tab. In the **Tag Show** message, click **OK**.
 - Click the **Factory Tags** or **Manual Tags** tab, select multiple tags, and then click **Classification** in the lower part of the tab. In the **Categorize Factory Tags** dialog box, enter a category name in the search box to search for your desired category and select the category. Then, click **OK**.

In the **Categorize Factory Tags** dialog box, you can also create a category by performing the following steps:

- a. Click **Create Category** in the upper-right corner.
- b. In the **Create Tag Category** dialog box, set the **Display Name**, **Name**, and **Description** parameters.

 **Note** The values of **Display Name** and **Name** cannot be numerical strings.

- c. Click **OK**.

9.11. Task publishing

9.11.1. Publishing management

In Dev-Prod mode, to schedule data modeling, data processing, and data integration tasks that are generated in the development environment, you must publish them to the production environment. This topic describes how to publish tasks and view the publishing records.

- In Dev-Prod mode, tasks rest on the **Objects to Publish** page after you submit them. You must publish them before you can schedule and manage them in the production development.
- In Basic mode, tasks are directly submitted to the production environment.

 **Note** The publishing feature is available only in the development environment.

Objects to Publish page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **R&D** in the top navigation bar. The Develop page appears.
3. Click **Publish** in the top navigation bar. The **Objects to Publish** page appears.

The **Objects to Publish** page displays tasks and files on four tabs: **Pipeline Script**, **Standards and Modeling**, **Batch Processing**, and **Stream Processing**. Tasks include stream processing tasks, batch processing tasks, and sync tasks. Files include resources, batch processing functions, stream processing functions, stream metatables, and stream processing templates.

On the **Objects to Publish** page, you can click the **Pipeline Script**, **Standards and Modeling**, **Batch Processing**, or **Stream Processing** tab to view objects to be published as required. The four tabs provide similar features. This section uses the **Standards and Modeling** tab as an example. You can filter objects by last submitter, last submission time, object type, and operation. You can also enter a keyword in the search box to search for objects whose names contain the keyword.

Parameter	Description
Object Name	The name of the object.
Object ID	The ID of the object.
Object Type	The type of the object.
Version	The version of the object.
Operation	The operation that was performed on the object. Valid values: <ul style="list-style-type: none"> ◦ Add ◦ Update ◦ Delete
Last Submitted By	The user who submitted the object last time and the submission time.
Actions	<ul style="list-style-type: none"> ◦ a. Click the  icon in the Actions column to publish the object. <p>You can also select multiple objects and click Publish in the lower part of the page to publish them at a time.</p> ◦ b. In the Publish dialog box, click OK to publish one or more objects. ◦ Click the  icon in the Actions column to view the publishing records of the object. ◦ Click the  icon in the Actions column to edit the object on the page that appears.

Publishing History page

In the left-side navigation pane, click **Publishing History**. On the **Publishing History** page, you can search for published objects and view the publishing status and details of a specific object. You can also click the Edit icon in the Actions column of an object to edit the object on the page that appears. You can filter objects by last publisher, last publishing time, object type, operation, and publishing status. You can also enter a keyword in the search box to search for objects whose names contain the keyword.

Parameter	Description
Publish Name	The publishing name of the object.
Publish ID	The publishing ID of the object.
Object Name	The name of the object.
Object ID	The ID of the object.
Object Type	The type of the object.
Node ID	The task ID of the object.
Version	The version of the object.
Operation	The operation that was performed on the object. Valid values: <ul style="list-style-type: none"> • Add • Update • Delete
Published By/Start Publishing At	The user who published the object and the publishing time.
Publishing Status/Finish Publishing At	The publishing status of the object and the time when the publishing was completed. Publishing states include: <ul style="list-style-type: none"> • Publish Failed • Published Successfully • Publishing
Actions	<ul style="list-style-type: none"> • Click Details. In the Publishing Details dialog box, view the publishing details. • Click the  icon in the Actions column to edit the object on the page that appears. • Click the  icon in the Actions column. On the Objects to Publish page, view the objects that failed to be published. Note that objects that failed to be published will rest on the Objects to Publish page. You can publish them again.

9.12. Scheduling center

9.12.1. Overview

Dataphin supports policy-based scheduling and management of code tasks that are generated by data modeling, coding, and data distilling. The scheduling and management feature includes global management, logical table management, and distilling management.

Scheduling page

The Scheduling page contains the Global Management tab. You can go to the Global Management tab by performing the following steps:

1. [Log on to the Dataphin console.](#)

2. Go to the **Global Management** tab by using one of the following methods:
 - If the project that you accessed last time is in **Basic** or **Prod** mode, use one of the following methods to go to the **Global Management** tab:
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the Scheduling page, click the  icon next to the project name in the upper-left corner, click the **Dev** tab, and then select a project. The **Global Management** tab appears by default.
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev** tab, and then select a project.
 - c. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.
 - If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Global Management** tab:
 - On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Global Management** tab appears by default.
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

In Dev-Prod mode, the scheduling tasks in projects in Dev mode are independent of those in the corresponding projects in Prod mode.

- In projects in Dev mode, the Scheduling page contains the **Global Management** and **Logical Table Management** tabs.
- The scheduling tasks in projects in Dev mode cannot automatically run. You must manually run the scheduling tasks to generate instances.
- In projects in Prod mode, the Scheduling page contains the **Global Management**, **Logical Table Management**, and **Monitoring** tabs. You can configure alert rules to monitor tasks only in projects in Prod mode.
- Dataphin generates and runs scheduling tasks in projects in Prod mode based on scheduling policies.
- You can click the  icon next to the project name in the upper-left corner to switch between the Scheduling page of a project in Dev mode and that of the corresponding project in Prod mode.

For example, if you click the  icon on the Scheduling page of the test_dev project, the Scheduling page of the test project appears.

In projects in Basic mode:

- The Scheduling page contains the **Global Management**, **Logical Table Management**, and **Monitoring** tabs.
- Dataphin generates and runs scheduling tasks in projects in Basic mode based on scheduling policies. You can configure alert rules to monitor tasks in projects in Basic mode.

Tasks

A task is an object that can be scheduled in the production environment after you submit and publish its code and configuration in the development environment. A task can be scheduled at specified intervals or manually triggered. Tasks are classified into the following types based on the way they are run:

- **Recurring tasks:** the tasks for which a recurrence is configured. The scheduling system of Dataphin schedules recurring tasks at specified intervals and generates corresponding recurring instances.
- **One-time tasks:** the tasks that are not automatically triggered by the scheduling system. You can manually run a one-time task as needed. Each time you run a one-time task, a one-time instance is generated.
- **Stream processing tasks:** the tasks for processing streaming data. You must manually run production or test instances for stream processing tasks.

The scheduling sequence of a task is determined based on the recurrence, dependency, and priority that are configured for the task.

- To apply the code or configuration update of a task to the instance generated the next day, you must update the code or configuration before 23:00 on the current day.
- When an instance runs, Dataphin reads the latest code and scheduling configuration of the corresponding task.
- You can specify a data timestamp within the range from 1970 to 2099 for Dataphin to generate retroactive data. Each scheduling instance is retained for up to two weeks.

Instances

An instance is generated each time a task is run. Instances can be generated in one of the following ways:

- Recurring instances are automatically and periodically generated when recurring tasks are scheduled by the scheduling system.
- Retroactive data generation instances are generated when you generate retroactive data for tasks.
- One-time instances are generated when you run one-time tasks.
- Stream processing instances are generated when you run production or test instances for stream processing tasks.

Logical tables

Logical table management involves logical table tasks and logical table task instances.

- The **Logical Table Tasks** section displays the tasks of logical tables.
- The **Logical Table Task Instances** section displays the instances of logical table tasks that are run and the states of these instances.

Distilling management

Distilling management involves tasks and instances. Distilling management allows you to manage tasks and instances related to behavior rules and tags from a business standpoint.

9.12.2. Global management

9.12.2.1. One-time tasks

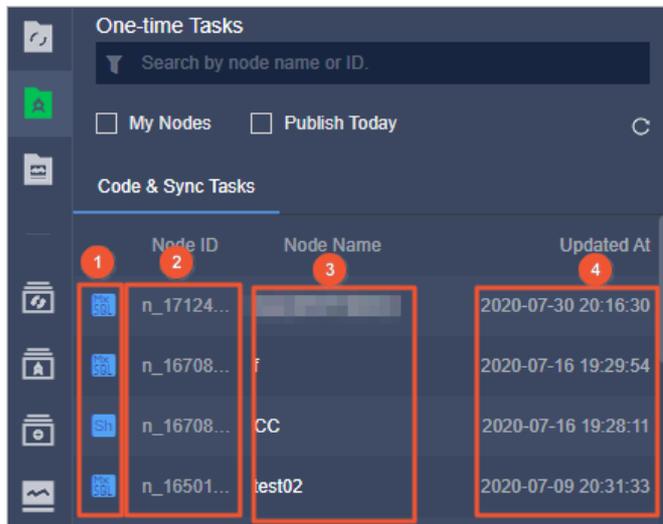
The One-time Tasks section on the Global Management tab displays the one-time tasks that have been submitted and published. This topic describes the operations that you can perform on one-time tasks. For example, you can view, run, and modify one-time tasks.

Go to the One-time Tasks section

To go to the One-time Tasks section, perform the following steps:

1. [Log on to the Dataphin console.](#)
2. Go to the **Global Management** tab.

- i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.
3. Click the  icon on the left-side navigation submenu. The **One-time Tasks** section appears.



No.	Description
1	The type of the task.
2	The node ID of the task.
3	The node name of the task.
4	The time when the task was last modified.

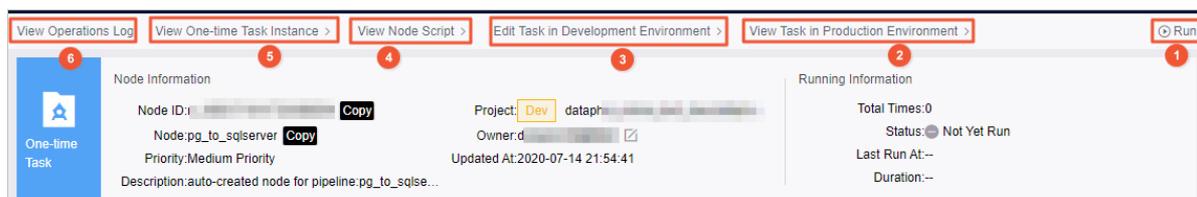
Search for one-time tasks

- Click the  icon in the search box, click the  icon next to **Owner** or **Task Type**, and then select an owner or a task type. You can also select both an owner and a task type to search for your desired one-time tasks.
- Enter a keyword in the search box to search for one-time tasks whose names contain the keyword.

Manage a one-time task in the right-side workspace

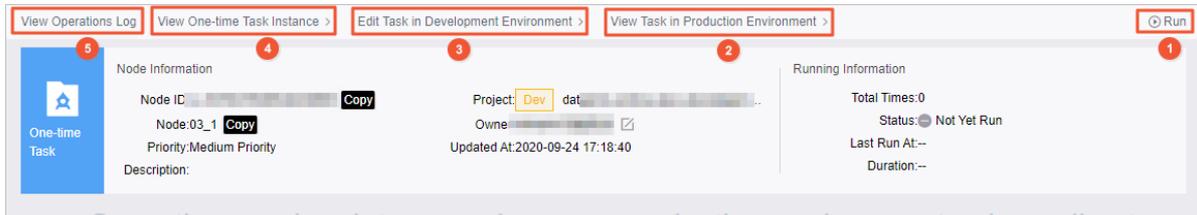
Click a one-time task in the **One-time Tasks** section. The task details appear in the right-side workspace.

- The following figure shows the workspace of a one-time task that is submitted or published from the **Integrated** module.



No.	Description
1	To run a one-time task, perform the following steps: <ol style="list-style-type: none"> i. Click Run in the upper-right corner. ii. In the Run dialog box, modify the Instance Name and Data Timestamp parameters or use their default values. iii. Click OK.
2	Click View Task in Production Environment to view the code of the task in the production environment.
3	Click Edit Task in Development Environment to edit the task in the development environment.
4	Click View Node Script to view the code of the task in the development environment.
5	Click View One-time Task Instance . In the One-time Task Instances section, view the instances that have been generated for the task.
6	Click View Operations Log . In the Operations Log pane, view the operation logs of the task.

- The following figure shows the workspace of a one-time task that is submitted or published from the **Develop** module.



No.	Description
1	To run a one-time task, perform the following steps: <ol style="list-style-type: none"> i. Click Run in the upper-right corner. ii. In the Run dialog box, modify the Instance Name and Data Timestamp parameters or use their default values. iii. Click OK.
2	Click View Task in Production Environment to view the code of the task in the production environment.
3	Click Edit Task in Development Environment to edit the task in the development environment.
4	Click View One-time Task Instance . In the One-time Task Instances section, view the instances that have been generated for the task.
5	Click View Operations Log . In the Operations Log pane, view the operation logs of the task.

9.12.2.2. Recurring tasks

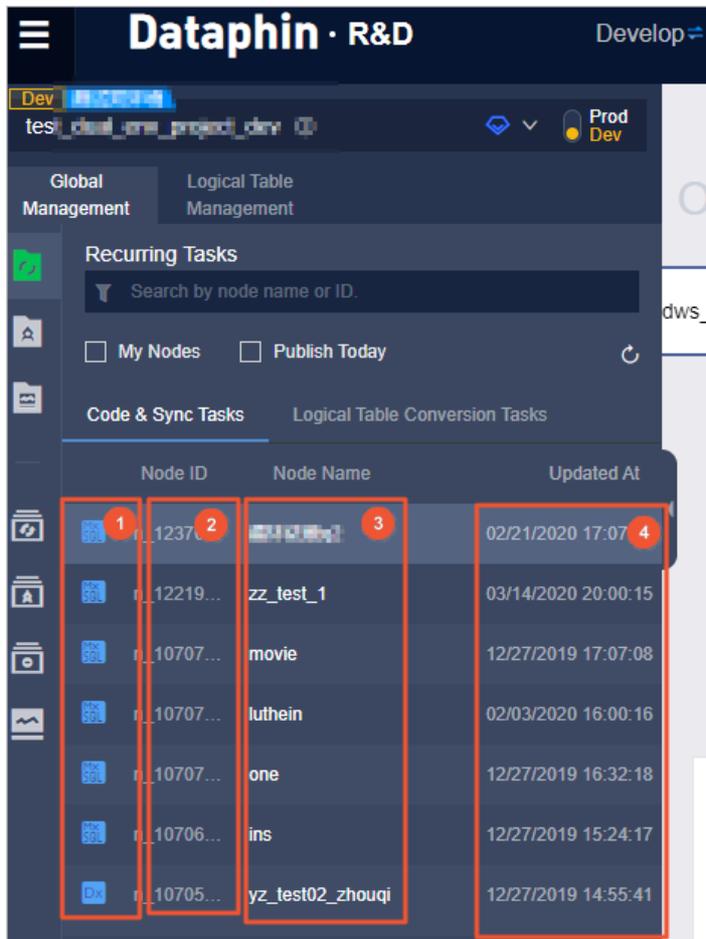
The Recurring Tasks section displays the recurring tasks that have been submitted and published. This topic describes the operations that you can perform on recurring tasks. For example, you can view recurring tasks and use DAGs to manage these tasks.

Go to the Recurring Tasks section

To go to the Recurring Tasks section, perform the following steps:

1. Log on to the Dataphin console.
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

Click the  icon on the left-side navigation submenu. The **Recurring Tasks** section appears.



No.	Description
1	The type of the task.
2	The node ID of the task.

No.	Description
3	The node name of the task.
4	The time when the task was last modified.

Search for recurring tasks

- Click the  icon in the search box, click the  icon next to **Owner** or **Task Type**, and then select an owner or a task type. You can also select both an owner and a task type to search for your desired recurring tasks.
- Enter a keyword in the search box to search for recurring tasks whose names contain the keyword.

Manage a recurring task in the right-side workspace

Click a recurring task in the **Recurring Tasks** section. The DAG of the task appears in the right-side workspace. Right-click the task node in the DAG. On the shortcut menu that appears, you can select menu items to perform supported operations.

- The following figure shows the shortcut menu for a recurring task that is submitted or published from the **Integrated** module.

Menu item	Description
Show Parent Nodes	Select Show Parent Nodes and select the layer that you want to view.
Show Child Nodes	Select Show Child Nodes and select the layer that you want to view.
View Node Script	Select View Node Script . On the Node Script page, view the code of the task.
View Script	Select View Script . On the Pipeline Details page, view the pipeline script of the task.
View Operations Log	Select View Operations Log . In the Operations Log pane, view the operation details of the task.
Change Owner	<p>Select Change Owner. In the Change Owner dialog box, select a member to whom you want to transfer the ownership of the task.</p> <div style="background-color: #e0f2f7; padding: 5px; border: 1px solid #ccc;"> <p> Note You can transfer a task only to a business unit administrator, the super administrator, or a project administrator.</p> </div>
View Instances	Choose View Instances > View Recurring Instances . In the Recurring Task Instances section, view the recurring instances that have been generated for the task.
	Choose View Instances > View Retroactive Data Generation Instances . In the Retroactive Data Generation Instances section, view the retroactive data generation instances that have been generated for the task.
Generate Retroactive Data	<p>To generate retroactive data for the task, perform the following steps:</p> <ol style="list-style-type: none"> i. Select Generate Retroactive Data. ii. In the Generate Retroactive Data dialog box, set the Instance Name, Data Timestamp, and Select Downstream Nodes parameters. iii. Click OK.

- The following figure shows the shortcut menu for a recurring task that is submitted or published from the **Develop** module.

Menu item	Description
Edit Node in Development Environment	Select Edit Node in Development Environment to edit the task in the development environment.
View Node in Production Environment	Select View Node in Production Environment to view the task in the production environment.
Show Parent Nodes	For more information, see the preceding table.
Show Child Nodes	
View Node Script	
View Operations Log	
Change Owner	
View Instances	Choose View Instances > View Recurring Instances . In the Recurring Task Instances section, view the recurring instances that have been generated for the task.
	Choose View Instances > View Retroactive Data Generation Instances . In the Retroactive Data Generation Instances section, view the retroactive data generation instances that have been generated for the task.
Generate Retroactive Data	To generate retroactive data for the task, perform the following steps: <ul style="list-style-type: none"> i. Select Generate Retroactive Data. ii. In the Generate Retroactive Data dialog box, set the Instance Name, Data Timestamp, and Select Downstream Nodes parameters. iii. Click OK.

9.12.2.3. Stream processing tasks

The Stream Processing Tasks section displays the stream processing tasks that have been submitted and published. These tasks must be run manually. This topic describes the operations that you can perform on stream processing tasks.

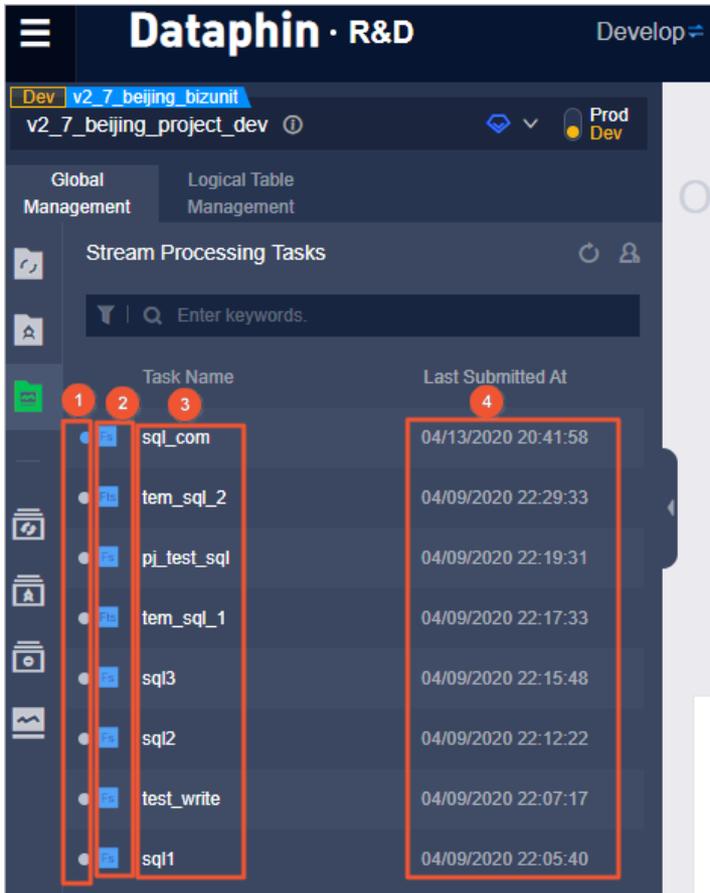
Go to the Stream Processing Tasks section

To go to the Stream Processing Tasks section, perform the following steps:

1. [Log on to the Dataphin console](#).
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the Dev, Basic, or Prod tab, and then select a project.
- iii. Click Scheduling in the top navigation bar. The Global Management tab appears by default.

Click the  icon on the left-side navigation submenu. The Stream Processing Tasks section appears.



No.	Description
1	<p>The status of the task.</p> <p>When you run a stream processing task in a project in Prod or Basic mode, a production instance is run.</p> <ul style="list-style-type: none"> • : A production instance is running. • : No production instance is running. <p>When you run a stream processing task in a project in Dev mode, a test instance is run.</p> <ul style="list-style-type: none"> • : A test instance is running. • : No test instance is running.

No.	Description
2	<p>The type of the task. Each task type is indicated by a unique icon.</p> <ul style="list-style-type: none"> : FLINK_SQL : FLINK_TEMPLATE_SQL
3	<p>The name of the task. The task name is the name of the stream processing task that you submitted.</p>
4	<p>The time when the task was last modified. By default, stream processing tasks are sorted in descending order of running time.</p>

Search for stream processing tasks

- In the Stream Processing Tasks section, click the  icon and then the  icon. In the drop-down list that appears, select **FLINK_SQL** or **FLINK_TEMPLATE_SQL** to filter tasks of the specified type.
- Enter a task name in the search box to search for the specific task. Alternatively, enter a keyword in the search box to search for tasks whose names contain the keyword.

View stream processing tasks

Click the  icon next to Stream Processing Tasks to view the stream processing tasks that you own.

Manage a stream processing task in the right-side workspace

Click a stream processing task in the **Stream Processing Tasks** section. The task details appear in the right-side workspace.

If a stream processing task in a project in Dev mode has no running test instance, click **Start Test Instance** in the upper-right corner of the workspace.

1. In the **Test Instance Parameter Configuration** dialog box, set the parameters as required.
2.
 - If you select **Print Log** for **Test Method**, set **Start Time for Reading Data**. Then, click **Start** to run a test instance.
 - If you select **Generate Table** for **Test Method**, set **Start Time for Reading Data** and specify the output table. Then, click **Start** to run a test instance.

The DAG displays information about the nodes of a stream processing task and relationships between the nodes. After you click a node in the DAG, the node details appear in the lower-left corner of the workspace.

You can perform the following operations on the task in the workspace:

- Click **Task Script** in the top navigation bar of the workspace. In the Task Scripts pane, view the code of the task. You can also click **Go to R&D Workbench and Edit Task Code** to edit the code.
- Click **Task Parameters** in the top navigation bar of the workspace. In the Task Parameters pane, view the parameters of the task. You can also click **Go to R&D Workbench and Edit Task Parameters** to modify the parameters.
- Click **Operations Log** in the top navigation bar of the workspace. In the Operations Log pane, view the operation logs of the task.
- ◦ If the task belongs to a project in Dev mode, click **View Test Instance** in the top navigation bar of the workspace to view the test instance in the **Stream Processing Instances** section.

- If the task belongs to a project in Prod or Basic mode, click **View Production Instances** in the top navigation bar of the workspace to view the production instance in the **Stream Processing Instances** section.
- If the task type is `FLINK_TEMPLATE_SQL`, click **Template Details** in the top navigation bar of the workspace. In the Template Details pane, view the template information. You can also click **Go to R&D Workbench to Edit Template** to edit the template information.

9.12.2.4. One-time instances

One-time instances are generated when you run one-time tasks manually. This topic describes the operations that you can perform on one-time instances.

Go to the One-time Task Instances section

To go to the One-time Task Instances section, perform the following steps:

1. [Log on to the Dataphin console](#).
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

Click the  icon on the left-side navigation submenu. The **One-time Task Instances** section appears.

No.	Description
1	The running status of the instance.
2	The task type of the instance. Each task type is indicated by a unique icon. <ul style="list-style-type: none">• : MAX_COMPUTE_SQL• : PYTHON• : SHELL• : MAX_COMPUTE_MR• : SPARK_JAR_ON_MAX_COMPUTE• : VIRTUAL
3	The node ID of the instance.
4	The node name of the instance.

Search for one-time instances

You can use one of the following methods to search for one-time instances:

- Click the  icon in the search box, click the  icon next to **Owner**, **Task Type**, or **Status**, and then select an owner, a task type, or a status. You can also specify a flexible combination of **Owner**, **Task Type**, and **Status** to search for your desired one-time instances.

- Enter an instance name in the search box to search for the specific instance. Alternatively, enter a keyword in the search box to search for instances whose names contain the keyword.

Manage a one-time instance in the right-side workspace

Click a one-time instance in the **One-time Task Instances** section. The instance details appear in the right-side workspace.

- The following figure shows the workspace of a one-time instance that is generated for a one-time task submitted or published from the **Integrated** module.

No.	Description
1	To rerun a one-time instance, perform the following steps: <ol style="list-style-type: none"> i. Click Rerun. ii. In the Rerun message, click OK.
2	Click View Task in Production Environment to view the code of the task corresponding to the instance in the production environment.
3	Click Edit Task in Development Environment to edit the task corresponding to the instance in the development environment.
4	Click View Node Script to view the code of the task corresponding to the instance in the development environment.
5	Click View One-time Task Instance . In the One-time Tasks section, view the task for which the instance was generated.
6	Click View Operational Log . On the Operational Log page, view the operational logs of the instance.

- The following figure shows the workspace of a one-time instance that is generated for a one-time task submitted or published from the **Develop** module.

No.	Description
1	To rerun a one-time instance, perform the following steps: <ol style="list-style-type: none"> i. Click Rerun. ii. In the Rerun message, click OK.
2	Click View Task in Production Environment to view the code of the task corresponding to the instance in the production environment.
3	Click Edit Task in Development Environment to edit the task corresponding to the instance in the development environment.
4	Click View One-time Task Instance . In the One-time Tasks section, view the task for which the instance was generated.
5	Click View Operational Log . On the Operational Log page, view the operational logs of the instance.

9.12.2.5. Recurring instances

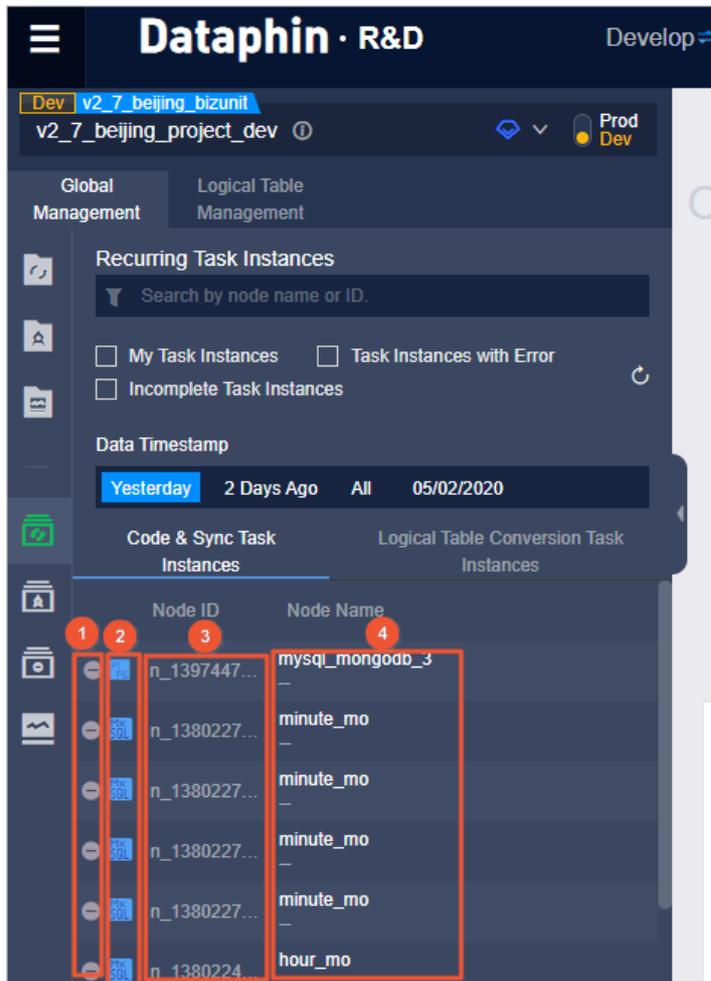
Recurring instances are generated when recurring tasks are periodically scheduled in the production environment. This topic describes the operations that you can perform on recurring instances. For example, you can view recurring instances and use DAGs to manage these instances.

Go to the Recurring Task Instances section

To go to the Recurring Task Instances section, perform the following steps:

1. [Log on to the Dataphin console.](#)
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.
- o If the project that you accessed last time is in **Basic** or **Prod** mode, use one of the following methods to go to the **Global Management** tab:
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the Scheduling page, click the  icon next to the project name in the upper-left corner, click the **Dev** tab, and then select a project.
 - c. The **Global Management** tab appears by default.
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev** tab, and then select a project.
 - c. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.
- o If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Global Management** tab:
 - On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Global Management** tab appears by default.
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

Click the  icon on the left-side navigation submenu. The **Recurring Task Instances** section appears.



No.	Description
1	The running status of the instance.
2	The task type of the instance. Each task type is indicated by a unique icon. <ul style="list-style-type: none"> : MAX_COMPUTE_SQL : MAX_COMPUTE_MR : SPARK_JAR_ON_MAX_COMPUTE : SHELL : PYTHON : VIRTUAL
3	The node ID of the instance.
4	The node name of the instance.

Search for recurring instances

You can use one of the following methods to search for recurring instances:

- Click the  icon in the search box, click the  icon next to **Owner**, **Task Type**, or **Status**, and then select an owner, a task type, or a status. You can also specify a flexible combination of **Owner**, **Task Type**, and **Status** to search for your desired recurring instances.
- Enter an instance name in the search box to search for the specific instance. Alternatively, enter a keyword in the search box to search for instances whose names contain the keyword.

Manage a recurring instance in the right-side workspace

Right-click an instance node in the DAG. On the shortcut menu that appears, you can select menu items to perform supported operations.

- The following figure shows the shortcut menu for a recurring instance that is generated for a recurring task submitted or published from the **Integrated** module.

Menu item	Description
Show Parent Nodes	Select Show Parent Nodes and select the layer that you want to view.
Show Child Nodes	Select Show Child Nodes and select the layer that you want to view.
View Operational Log	Select View Operational Log . On the Operational Log page, view the operational logs of the instance.
View Node Script	Select View Node Script . On the Node Script page, view the code of the instance.
View Script	Select View Script . On the Pipeline Details page, view the pipeline script of the instance.
View Operations Log	Select View Operations Log . In the Operations Log pane, view the operation details of the instance.
Terminate	Select Terminate to terminate the instance if it is in the Wait Submission or Running state.
Rerun and Resume Scheduling	Select Rerun and Resume Scheduling to rerun the instance if it is in the Success , Not Running , or Failed state.
Rerun Downstream Nodes	Select Rerun Downstream Nodes to rerun the downstream nodes of the instance.
Set to Succeeded and Resume Scheduling	Select Set to Succeeded and Resume Scheduling to set the running status of the instance to Success and schedule the instance.
Remove Upstream Dependency	Select Remove Upstream Dependency to remove the upstream dependencies of the instance.
Force Rerun	Select Force Rerun to forcibly rerun the instance.
Pause	Select Pause to pause the instance.

- The following figure shows the shortcut menu for a recurring instance that is generated for a recurring task submitted or published from the **Develop** module.
 - Click **Edit Node in Development Environment** to edit the task corresponding to the instance in the development environment.

- o Click **View Node in Production Environment** to view the task corresponding to the instance in the production environment.

For more information about other supported operations, see the preceding table.

9.12.2.6. Stream processing instances

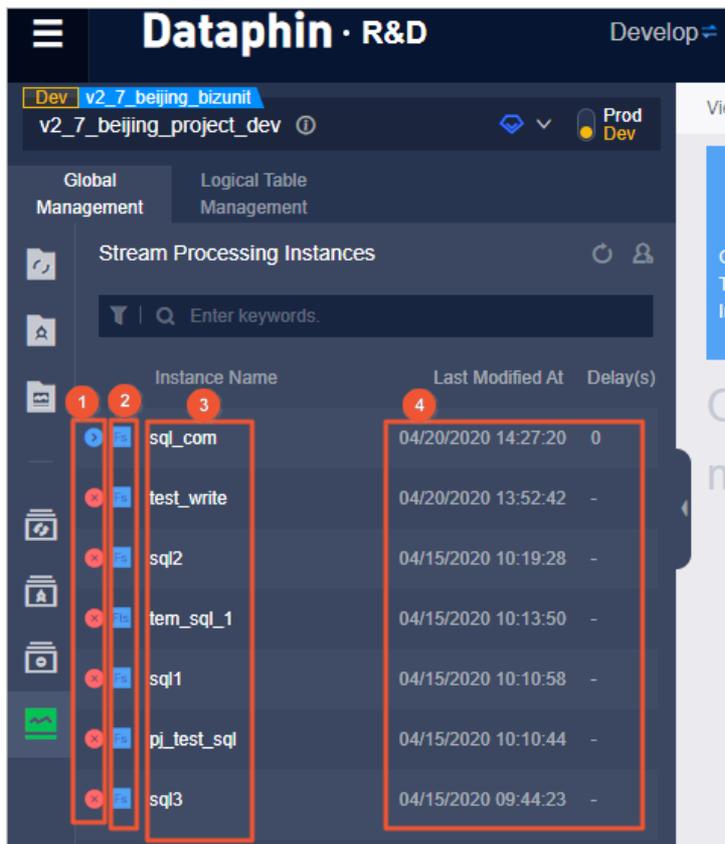
When you run a stream processing task in Dataphin, an instance is generated. A stream processing task has only one instance in an environment. This topic describes the operations that you can perform on stream processing instances.

Go to the Stream Processing Instances section

To go to the Stream Processing Instances section, perform the following steps:

1. [Log on to the Dataphin console](#)
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

Click the  icon on the left-side navigation submenu. The **Stream Processing Instances** section appears.



No.	Description
-----	-------------

No.	Description
1	<p>The running status of the instance. Each running status is indicated by a unique icon.</p> <ul style="list-style-type: none">  Running  Terminate  Terminating  Suspend  Suspending  Starting  Failed to Run <p>After you resume a paused instance, the instance status changes from Suspend to Resuming and the icon changes to .</p>
2	<p>The task type of the instance. Each task type is indicated by a unique icon.</p> <ul style="list-style-type: none">  FLINK_SQL  FLINK_TEMPLATE_SQL
3	<p>The name of the instance. The instance name is the name of the submitted stream processing task.</p>
4	<p>The time when the instance was last modified. By default, stream processing instances are sorted in descending order of running time.</p>

Search for stream processing instances

- In the Stream Processing Instances section, click the  icon and then the  icon. In the drop-down list that appears, select **FLINK_SQL** or **FLINK_TEMPLATE_SQL** to filter instances of the specified type.
- Enter an instance name in the search box to search for the specific instance. Alternatively, enter a keyword in the search box to search for instances whose names contain the keyword.

View stream processing instances

Click the  icon next to Stream Processing Instances to view the stream processing instances that you own.

Manage a stream processing instance in the right-side workspace

Click a stream processing instance in the **Stream Processing Instances** section. The DAG of the instance appears in the right-side workspace. The DAG shows the relationships between the nodes of the stream processing instance and the resource consumption of the nodes. After you click a node in the DAG, the node details appear in the lower-left corner of the workspace.

You can perform the following operations on the instance in the workspace:

- Click **Instance Code** in the top navigation bar of the workspace. In the Instance Code pane, view the code of the instance. You can also click **Go to R&D Workbench and Edit Task Code** to edit the code.

- Click **Instance Parameters** in the top navigation bar of the workspace. In the Instance Parameters pane, view the parameters of the instance. You can also click **Go to R&D Workbench and Edit Task Parameters** to modify the parameters.
- Click **Operations Log** in the top navigation bar of the workspace. In the Operations Log pane, view the operation logs of the instance.
- Click **View Stream Processing Task** in the top navigation bar of the workspace. In the Stream Processing Tasks section, view the stream processing task for which the instance was generated.
- If the task type is `FLINK_TEMPLATE_SQL`, click **Template Details** in the top navigation bar of the workspace. In the Template Details pane, view the template information. You can also click **Go to R&D Workbench to Edit Template** to edit the template information.
- Click **Runtime Analysis** in the top navigation bar of the workspace. On the page that appears, view the running details of the instance, such as the failover information and logs.

In projects in Basic or Prod modes, you can pause, terminate, resume, rerun, and unpublish production instances. In projects in Dev mode, you can pause, rerun, and unpublish test instances. For more information, see the following description:

- For an instance in the **Running** state, you can click **Suspend** or **Terminate** in the upper-right corner of the workspace to pause or terminate the instance.
- For an instance in the **Suspend** state, you can click **Resume** or **Terminate** in the upper-right corner of the workspace to resume or terminate the instance.
- For an instance in the **Failed to Run** state, you can click **Rerun** or **Unpublish** in the upper-right corner of the workspace to rerun or delete the instance.

9.12.2.7. Retroactive data generation instances

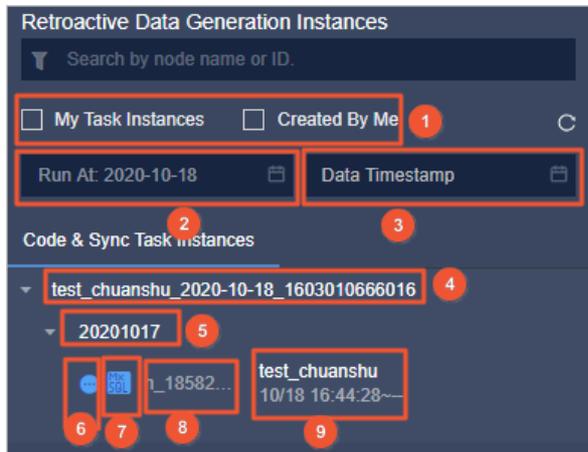
Retroactive data generation instances are generated when you generate retroactive data for recurring tasks. This topic describes the operations that you can perform on retroactive data generation instances.

Go to the Retroactive Data Generation Instances section

To go to the Retroactive Data Generation Instances section, perform the following steps:

1. [Log on to the Dataphin console](#).
2. Go to the **Global Management** tab.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Global Management** tab appears by default.

Click the  icon in the left-side navigation submenu. The **Retroactive Data Generation Instances** section appears. By default, this section displays the retroactive data generation instances that were run on the current day in descending order of running time.



No.	Description
1	The conditions for filtering the retroactive data generation instances that you own and those you created. You can select My Task Instances to filter the instances that you own or Created By Me to filter the instances that you created.
2	The date on which the retroactive data generation instance was run. You can click Run At and select a date to filter the instances that were run on that day.
3	The data timestamp of the retroactive data generation instance. You can click Data Timestamp and select a date to filter the instances that generate data for that date.
4	The display name of the retroactive data generation instance. The display name is in the format <code>Name of the task corresponding to the instance_Date on which the instance was run_Name of the instance</code> .
5	The data timestamp of the retroactive data generation instance. The data timestamp is the date for which the retroactive data generation instance generates data required by the corresponding recurring task.
6	The running status of the retroactive data generation instance.
7	The type of the recurring task for which the retroactive data generation instance was generated.
8	The node ID of the retroactive data generation instance.
9	The node name of the retroactive data generation instance.

Search for retroactive data generation instances

You can use one of the following methods to search for retroactive data generation instances:

- Click the  icon in the search box, click the icon next to **Owner** or **Status**, and then select an owner or a status. You can also select both an owner and a status to search for your desired retroactive data generation instances.
- Enter the name of a retroactive data generation instance in the search box to search for the specific

instance. Alternatively, enter a keyword in the search box to search for retroactive data generation instances whose names contain the keyword.

Manage a retroactive data generation instance in the right-side workspace

Click a retroactive data generation instance in the **Retroactive Data Generation Instances** section. The DAG of the instance appears in the right-side workspace. In the DAG, you can perform the following operations:

- Right-click the root node. On the shortcut menu that appears, select menu items to perform supported operations.

Menu item	Description
Show Parent Nodes	Select Show Parent Nodes and select the layer that you want to view.
Show Child Nodes	Select Show Child Nodes and select the layer that you want to view.
View Operational Log	Select View Operational Log . On the Operational Log page, view the operational logs of the root node.

- Right-click the instance node. On the shortcut menu that appears, select menu items to perform supported operations.

Menu item	Description
Show Parent Nodes	Select Show Parent Nodes and select the layer that you want to view.
Show Child Nodes	Select Show Child Nodes and select the layer that you want to view.
View Operational Log	Select View Operational Log . On the Operational Log page, view the operational logs of the instance.
View Node Script	Select View Node Script . On the Node Script page, view the code of the task corresponding to the instance in the development environment.
Edit Node in Development Environment	Select Edit Task in Development Environment to edit the task corresponding to the instance in the development environment.
View Node in Production Environment	Select View Task in Production Environment to view the code of the task corresponding to the instance in the production environment.
View Operations Log	Select View Operations Log . In the Operations Log pane, view the operation details of the instance.
Terminate	Select Terminate to terminate the instance if it is in the Wait Submission or Running state.
Rerun and Resume Scheduling	Select Rerun and Resume Scheduling to rerun the instance if it is in the Success , Not Running , or Failed state.
Rerun Downstream Nodes	Select Rerun Downstream Nodes to rerun the downstream nodes of the instance.

Menu item	Description
Set to Succeeded and Resume Scheduling	Select Set to Succeeded and Resume Scheduling to set the running status of the instance to Success and schedule the instance.
Remove Upstream Dependency	Select Remove Upstream Dependency to remove the upstream dependencies of the instance.
Force Rerun	Click Force Rerun to forcibly rerun the instance.
Pause	Click Pause to pause the instance.

9.12.3. Logical tables

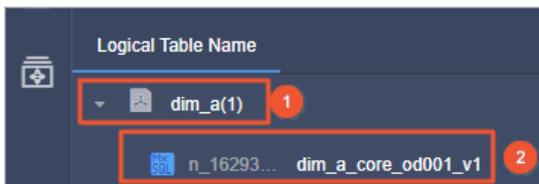
9.12.3.1. Logical table tasks

This topic describes the operations that you can perform on logical table tasks.

Go to the Logical Table Tasks section

1. [Log on to the Dataphin console.](#)
2. Go to the **Logical Table Tasks** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - ii. Click the  icon next to the project name, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - iv. Click the **Logical Table Management** tab. The **Logical Table Tasks** section appears.

In the **Logical Table Tasks** section, you can view and search for logical tables and their conversion tasks. If you click a logical table in the Logical Table Tasks section, the DAG of the logical table appears in the right-side workspace. The DAG shows all the tasks of the logical table and their relationships. The thumbnail of the DAG appears in the lower-right corner of the workspace.



No.	Description
1	The logical table.
2	The logical table tasks.

View fields

1. In the **Logical Table Tasks** section, click the logical table for which you want to view the fields.
2. Move the pointer over a logical table task and click the  icon.
3. In the Logical Table Conversion Task Node panel, view the fields that are contained in the logical table task.

View the code of a task

1. In the **Logical Table Tasks** section, click the logical table for which you want to view the code of a logical table task. In the DAG in the right-side workspace, right-click the node that corresponds to the task for which you want to view the code and select **View Node Script**.
2. On the **Node Script** page, view the code of the task.

View the global DAG of a task

1. In the **Logical Table Tasks** section, click the logical table task for which you want to view the global DAG. In the DAG in the right-side workspace, right-click the main node and select **View Recurring Task**.
2. On the **Global Management** tab, view the logical table task in the **Recurring Tasks** section and its global DAG in the right-side workspace.

View the global instance DAG of a task

1. In the **Logical Table Tasks** section, click the logical table task for which you want to view the global instance DAG. In the DAG in the right-side workspace, right-click the main node and choose **View Instances > View Recurring Instances**.
2. On the **Global Management** tab, view the instance of the logical table task in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

View retroactive data generation instances

1. In the **Logical Table Tasks** section, click the logical table task for which you want to view the retroactive data generation instances. In the DAG in the right-side workspace, right-click the node that corresponds to the task for which you want to view the retroactive data generation instances and choose **View Instances > View Retroactive Data Generation Instances**.
2. On the **Global Management** tab, view the retroactive data generation instances that are generated for the logical table task in the **Retroactive Data Generation Instances** section.

Generate retroactive data

1. In the **Logical Table Tasks** section, click the logical table task for which you want to generate retroactive data. In the DAG in the right-side workspace, right-click the node for which you want to generate retroactive data and select **Generate Retroactive Data**.
2. In the **Generate Retroactive Data** dialog box, set the **Instance Name**, **Data Timestamp**, and **Select Downstream Nodes** parameters.

If you set the **Select Downstream Nodes** parameter to **Yes**, you must select downstream nodes for which you want to generate retroactive data as prompted.

3. Click **OK**.

9.12.3.2. Logical table task instances

Logical table task instances are generated when logical table tasks are run. This topic describes the operations that you can perform on logical table task instances.

Go to the Logical Table Task Instances section

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Table Task Instances** section.
 - i. On the Dataphin homepage, click **R&D** in the top navigation bar.

- ii. On the Develop page, click the  icon next to the project name in the upper-left corner, click the **Dev**, **Basic**, or **Prod** tab, and then select a project.
 - iii. Click **Scheduling** in the top navigation bar. The **Scheduling** page appears.
 - iv. In the left-side navigation pane, click the **Logical Table Management** tab. The **Logical Table Tasks** section appears by default.
 - v. On the left-side navigation submenu, click the  icon. The **Logical Table Task Instances** section appears.
3. In the **Logical Table Task Instances** section, you can view and search for logical tables and their conversion task instances. If you click a logical table in the Logical Table Task Instances section, the DAG of the logical table appears in the right-side workspace. The DAG shows all the task instance nodes of the logical table and their running status, for example, Running, Success, or Failed. The thumbnail of the DAG appears in the lower-right corner of the workspace. Logical table task instances are displayed in a two-level hierarchy.



No.	Description
1	The logical table.
2	The logical table task instances.

View the fields that are contained in a logical table task instance

1. In the **Logical Table Task Instances** section, click the target logical table.
2. Move the pointer over the target logical table task instance and click the  icon.
3. In the Logical Table Conversion Task Node pane, view the fields that are contained in the logical table task instance.

View the operational logs of a logical table task instance

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and select **View Operational Log**.
2. On the **Operational Log** page, view the operational logs of the logical table task instance.

View the code of a logical table task instance

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and select **View Node Script**.

2. On the **Node Script** page, view the code of the logical table task instance.

Rerun a logical table task instance

 **Note** You can rerun a logical table task instance only when the instance is in the **Success**, **Not Running**, or **Failed** state.

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and select **Rerun and Resume Scheduling**.
2. In the **Rerun and Resume Scheduling** message, click **OK**.

View the global DAG of the logical table task for which a logical table task instance was generated

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and select **View Recurring Task**.
2. On the **Global Management** tab, view the task for which the logical table task instance was generated in the **Recurring Tasks** section and the global DAG of the task in the right-side workspace.

View the global DAG of a logical table task instance

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and choose **View Instances > View Recurring Instances**.
2. On the **Global Management** tab, view the logical table task instance in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

View the retroactive data generation instances that have been generated for the task corresponding to a logical table task instance

1. In the **Logical Table Task Instances** section, click the target logical table. In the DAG in the right-side workspace, right-click the node of the target logical table task instance and choose **View Instances > View Retroactive Data Generation Instances**.
2. On the **Global Management** tab, view the retroactive data generation instances that have been generated for the task corresponding to the logical table task instance in the **Retroactive Data Generation Instances** section.

9.12.4. Distilling management

9.12.4.1. Behavior rule tasks

Behavior rule tasks are generated when you submit behavior rules. This topic describes the operations that you can perform on behavior rule tasks.

Go to the Behavior Rule Tasks section

1. [Log on to the Dataphin console](#).
2. Go to the **Distilling Management** tab.
 - If the project that you accessed last time is the **Data_distill** project, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner of the Dataphin homepage.

- If the project that you accessed last time is not the **Data_distill** project, use one of the following methods to go to the **Distilling Management** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - c. On the **Distilling** page, click **Scheduling** in the top navigation bar.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Scheduling** page appears.
 - b. Click the  icon next to the project name in the upper-left corner, click the **Basic** tab, and then select the **Data_distill** project.
- 3. On the **Distilling Management** tab, the **Behavior Rule Tasks** section appears by default.
- 4. View behavior rule tasks in the **Behavior Rule Tasks** section.

You can view and search for behaviors, behavior rules, and behavior rule tasks in the Behavior Rule Tasks section. If you click a behavior rule task in the Behavior Rule Tasks section, the DAG of the task appears in the right-side workspace. The DAG shows the node of the current behavior rule task and its upstream and downstream nodes. The thumbnail of the DAG appears in the lower-right corner of the workspace. Behavior rule tasks are displayed in a three-level hierarchy.

No.	Description
1	The behavior.
2	The behavior rule.
3	The behavior rule task. You can move the pointer over the task to view the task information, including the task ID, name, owner, and submission time.

Search for behavior rule tasks

In the **Behavior Rule Tasks** section, you can search for behavior rule tasks.

You can enter a behavior name or behavior rule ID in the search box to search for a specific behavior rule task.

Modify a behavior rule

1. Go to the **Change Behavior Rules** tab by using one of the following methods:
 - In the **Behavior Rule Tasks** section, click the target behavior. Move the pointer over the  icon next to the target behavior rule and select **Modify Behavior Rules**.
 - In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the Behavior Rule box and select **Modify Behavior Rules**.
2. On the **Change Behavior Rules** tab, modify the behavior rule. For more information, see [Edit a behavior rule](#).

View the task instances of a behavior rule

1. Go to the **Behavior Rule Task Instances** section by using one of the following methods:
 - In the **Behavior Rule Tasks** section, click the target behavior. Move the pointer over the  icon next to the target behavior rule and select **View Behavior Rule Instances**.
 - In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In

the DAG in the right-side workspace, right-click the Behavior Rule box and choose **View Instance > View Behavior Rule Instances**.

2. In the **Behavior Rule Task Instances** section, view the task instances that have been generated for the behavior rule. You can view the details of each task instance.

Generate retroactive data for a single behavior rule

1. Open the **Generate Retroactive Data for Behavior Rules** dialog box by using one of the following methods:
 - In the **Behavior Rule Tasks** section, click the target behavior. Move the pointer over the  icon next to the target behavior rule and select **Generate Retroactive Data for Behavior Rules**.
 - In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the Behavior Rule box and select **Generate Retroactive Data for Behavior Rules**.
2. In the **Generate Retroactive Data for Behavior Rules** dialog box, set the parameters in the **Effective Period** and **Instance Name** sections.
3. Click **OK**.

Generate retroactive data for multiple behavior rules under a behavior

1. In the **Behavior Rule Tasks** section, move the pointer over the  icon next to the target behavior and select **Generate Retroactive Data**.
2. In the **Generate Retroactive Data for Behavior Rules** dialog box, select the behavior rules for which you want to generate retroactive data and set the parameters in the **Effective Period** and **Instance Name** sections.
3. Click **OK**.

View the retroactive data generation instances for a behavior rule

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the Behavior Rule box and choose **View Instance > View Retroactive Data Generation Instance of Behavior Rule**.
2. In the **Retroactive Data Generation Instances** section, view the retroactive data generation instances that have been generated for the behavior rule. You can view the details of each retroactive data generation instance.

View the code of a behavior rule task

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the task node and select **View Node Script**.
2. On the **Node Script** page, view the code of the task.

View the operation logs of a behavior rule task

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the task node and select **View Operations Log**.
2. In the **Operations Log** pane, view the operation logs of the task. You can click the  icon in the **Details** column to view the operation details.

View the global DAG of a behavior rule task

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In

the DAG in the right-side workspace, right-click the task node and select **View Global Tasks**.

2. On the **Global Management** tab, view the behavior rule task in the **Recurring Tasks** section and the global DAG of the task in the right-side workspace.

View the global instance DAG of a behavior rule task

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the task node and select **View Global Task Instances**.
2. On the **Global Management** tab, view the instance of the behavior rule task in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

Generate retroactive data for a behavior rule task

1. In the **Behavior Rule Tasks** section, click the target behavior and then the target behavior rule task. In the DAG in the right-side workspace, right-click the task node and select **Generate Retroactive Data**.
2. In the **Generate Retroactive Data** dialog box, set the **Instance Name**, **Data Timestamp**, and **Select Downstream Nodes** parameters.

If you set **Select Downstream Nodes** to **Yes**, you must select downstream nodes for which you want to generate retroactive data.

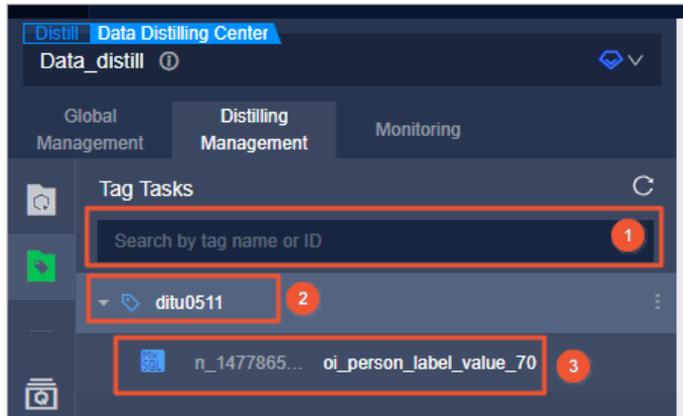
3. Click **OK**.

9.12.4.2. Tag tasks

This topic describes the operations that you can perform on tags and tag tasks. For example, you can edit a tag, generate retroactive data for a tag, and view the task instances of a tag.

Go to the Tag Tasks section

1. [Log on to the Dataphin console](#).
2. Go to the **Distilling Management** tab.
 - If the project that you accessed last time is the **Data_distill** project, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner of the Dataphin homepage.
 - If the project that you accessed last time is not the **Data_distill** project, use one of the following methods to go to the **Distilling Management** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - c. On the **Distilling** page, click **Scheduling** in the top navigation bar.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Scheduling** page appears.
 - b. Click the  icon next to the project name in the upper-left corner, click the **Basic** tab, and then select the **Data_distill** project.
3. On the left-side navigation submenu, click the  icon.
4. In the **Tag Tasks** section, view tag tasks.



No.	Description
1	The search box for you to search for tag tasks by tag name or ID.
2	The tag.
3	The tag task. The display name of a tag task is in the format of Tag task type_Tag task ID_Tag task name.

Generate retroactive data for a tag

- Open the **Generate Retroactive Data for Tags** dialog box by using one of the following methods:
 - In the **Tag Tasks** section, move the pointer over the  icon next to the target tag and select **Generate Retroactive Data for Tags**.
 - In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the Tag box and select **Generate Retroactive Data for Tags**.
- In the **Generate Retroactive Data for Tags** dialog box, set the parameters for the retroactive data generation instance.

Section	Operation
Effective Period	Specify the end date of the period for which you want to generate retroactive data.
Instance Name	Set the name of the retroactive data generation instance to be generated for the tag.

- Click **OK**.

Edit a tag

- Go to the **Change Tag** tab by using one of the following methods:
 - In the **Tag Tasks** section, move the pointer over the  icon next to the target tag and select **Edit Tag**.
 - In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the Tag box and select **Edit Tag**.
- On the **Change Tag** tab, modify the tag parameters as required. For more information, see [Edit a factory tag](#).

View the task instances of a tag

1. Go to the **Tag Task Instances** section by using one of the following methods:
 - In the **Tag Tasks** section, move the pointer over the  icon next to the target tag and select **View Tag Instances**.
 - In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the Tag box and choose **View Instance > Tag Instance**.
2. In the **Tag Task Instances** section, view the task instances that have been generated for the tag.

View the retroactive data generation instances for a tag

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the Tag box and choose **View Instance > Retroactive Data Generation Instance for Tags**.
2. In the **Retroactive Data Generation Instances** section, view the retroactive data generation instances that have been generated for the tag. You can view the details of each retroactive data generation instance.

View the code of a tag task

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the task node and select **View Node Script**.
2. On the **Node Script** page, view the code of the task.

View the operation logs of a tag task

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the task node and select **View Operations Log**.
2. In the **Operations Log** pane, view the operation logs of the task. You can click the  icon in the **Details** column to view the operation details.

View the global DAG of a tag task

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the task node and select **View Global Tasks**.
2. On the **Global Management** tab, view the tag task in the **Recurring Tasks** section and the global DAG of the task in the right-side workspace.

View the global instance DAG of a tag task

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the task node and select **View Global Task Instances**.
2. On the **Global Management** tab, view the instance of the tag task in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

Generate retroactive data for a tag task

1. In the **Tag Tasks** section, click the target tag and then the target tag task. In the DAG in the right-side workspace, right-click the task node and select **Generate Retroactive Data**.
2. In the **Generate Retroactive Data** dialog box, set the **Instance Name**, **Data Timestamp**, and **Select Downstream Nodes** parameters.

If you set **Select Downstream Nodes** to **Yes**, you must select downstream nodes for which you want to generate retroactive data.

3. Click **OK**.

9.12.4.3. Behavior rule task instances

Behavior rule task instances are generated when behavior rule tasks are run. This topic describes the operations that you can perform on behavior rule task instances.

Go to the Behavior Rule Task Instances section

1. [Log on to the Dataphin console](#).
2. Go to the **Distilling Management** tab.
 - If the project that you accessed last time is the **Data_distill** project, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner of the Dataphin homepage.
 - If the project that you accessed last time is not the **Data_distill** project, use one of the following methods to go to the **Distilling Management** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - c. On the **Distilling** page, click **Scheduling** in the top navigation bar.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Scheduling** page appears.
 - b. Click the  icon next to the project name in the upper-left corner, click the **Basic** tab, and then select the **Data_distill** project.
3. On the left-side navigation submenu, click the  icon.
4. In the **Behavior Rule Task Instances** section, view behavior rule task instances.

You can view and search for behaviors, behavior rules, and behavior rule task instances in the Behavior Rule Task Instances section. If you click a behavior rule task instance in the Behavior Rule Task Instances section, the DAG of the instance appears in the right-side workspace. The DAG shows the node of the current behavior rule task instance and its upstream and downstream nodes. The thumbnail of the DAG appears in the lower-right corner of the workspace. Behavior rule task instances are displayed in a three-level hierarchy.



No.	Description
-----	-------------

No.	Description
1	<p>The behavior and the overall running status of all task instances under the behavior.</p> <ul style="list-style-type: none">◦ The overall running status is Success only if all task instances under the behavior are properly run.◦ The overall running status is Failed if any task instance under the behavior failed.◦ The overall running status is Running if some task instances under the behavior are properly run and the others are running.
2	<p>The behavior rule.</p>
3	<p>The behavior rule task instances. You can move the pointer over an instance to view the instance information, including the instance ID, name, owner, and running time.</p>

Search for behavior rule task instances

In the **Behavior Rule Task Instances** section, you can search for behavior rule task instances by using one of the following methods:

- Click **Yesterday** or **2 Days Ago** below **Data Timestamp** to filter behavior rule task instances.
- Enter a behavior name or behavior rule ID in the search box to search for specific behavior rule task instances.
- Click the date below **Data Timestamp** and select a date to filter behavior rule task instances.

Modify a behavior rule

1. Go to the **Change Behavior Rules** tab by using one of the following methods:
 - In the **Behavior Rule Task Instances** section, click the target behavior. Move the pointer over the  icon next to the target behavior rule and select **Modify Behavior Rules**.
 - In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the **Behavior Rule** box and select **Modify Behavior Rules**.
2. On the **Change Behavior Rules** tab, modify the behavior rule. For more information, see [Edit a behavior rule](#).

View the behavior rule task under a behavior rule

1. Go to the **Behavior Rule Tasks** section by using one of the following methods:
 - In the **Behavior Rule Task Instances** section, click the target behavior. Move the pointer over the  icon next to the target behavior rule and select **View Behavior Rule Tasks**.
 - In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the **Behavior Rule** box and select **View Behavior Rule Tasks**.
2. In the **Behavior Rule Tasks** section, view the behavior rule task under the behavior rule. You can view the details of the task.

View the operational logs of a behavior rule task instance

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule

task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **View Operational Log**.

2. On the **Operational Log** page, view the operational logs of the instance.

View the code of a behavior rule task instance

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **View Node Script**.
2. On the **Node Script** page, view the code of the instance.

View the operation logs of a behavior rule task instance

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **View Operations Log**.
2. In the **Operations Log** pane, view the operation logs of the instance. You can click the  icon in the **Details** column to view the operation details.

View the global DAG of the behavior rule task for which a behavior rule task instance was generated

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **View Global Tasks**.
2. On the **Global Management** tab, view the behavior rule task for which the instance was generated in the **Recurring Tasks** section and the global DAG of the task in the right-side workspace.

View the global DAG of a behavior rule task instance

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **View Global Task Instances**.
2. On the **Global Management** tab, view the instance in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

Terminate a behavior rule task instance

 **Note** You can terminate a behavior rule task instance only when the instance is in the **Wait Submission** or **Running** state.

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **Terminate**.
2. In the **Terminate** message, click **OK**.

Rerun a behavior rule task instance

 **Note**

- You can rerun a behavior rule task instance only when the instance is in the **Success**, **Not Running**, or **Failed** state.
- Use caution when you rerun a behavior rule task instance, because the data generated by the instance previously will be overwritten after the instance is rerun.

1. In the **Behavior Rule Task Instances** section, click the target behavior, behavior rule, and behavior rule task instance in sequence. In the DAG in the right-side workspace, right-click the instance node and select **Rerun and Resume Scheduling**.
2. In the **Rerun and Resume Scheduling** message, click **OK**.

9.12.4.4. Tag task instances

Tag task instances are generated when tag tasks are run. This topic describes the operations that you can perform on tag task instances.

Go to the Tag Task Instances section

1. Log on to the [Dataphin console](#).
2. Go to the **Distilling Management** tab.
 - If the project that you accessed last time is the **Data_distill** project, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner of the Dataphin homepage.
 - If the project that you accessed last time is not the **Data_distill** project, use one of the following methods to go to the **Distilling Management** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - c. On the **Distilling** page, click **Scheduling** in the top navigation bar.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Scheduling** page appears.
 - b. Click the  icon next to the project name in the upper-left corner, click the **Basic** tab, and then select the **Data_distill** project.
3. On the left-side navigation submenu, click the  icon.
4. In the **Tag Task Instances** section, view tag task instances.

You can view and search for tags and tag task instances in the Tag Task Instances section. If you click a tag task instance in the Tag Task Instances section, the DAG of the instance appears in the right-side workspace. The DAG shows the node of the current tag task instance and its upstream and downstream nodes. The thumbnail of the DAG appears in the lower-right corner of the workspace.

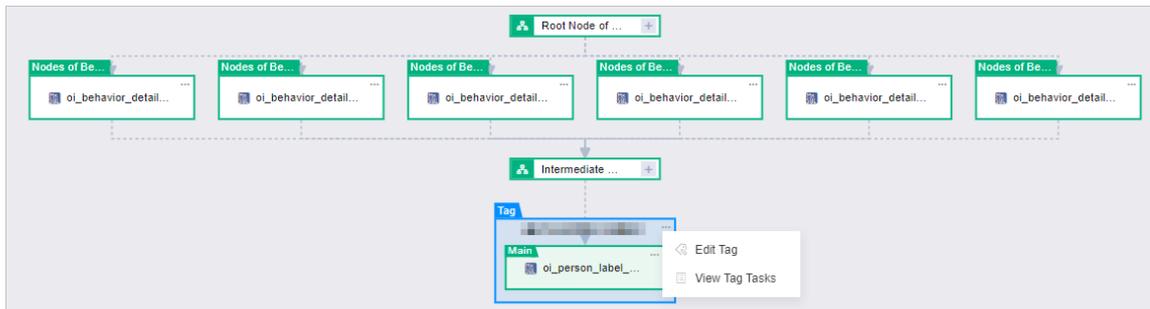
Search for tag task instances

In the **Tag Task Instances** section, you can search for tag task instances by using one of the following methods:

- Enter a tag name or tag ID in the search box to search for specific tag task instances.
- Click **Yesterday** or **2 Days Ago** or select a date below **Data Timestamp** to filter tag task instances.

Edit a tag

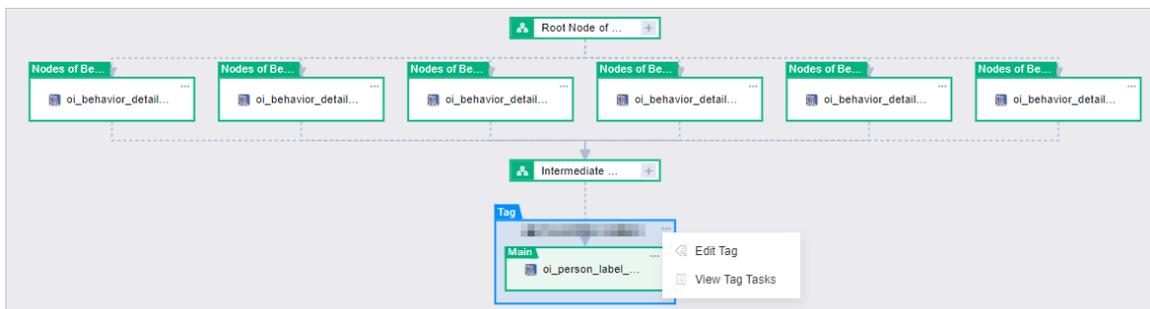
- Go to the Change Tag tab by using one of the following methods:
 - In the **Tag Task Instances** section, move the pointer over the  icon next to the target tag and select **Edit Tag**.
 - In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the Tag box and select **Edit Tag**.



- On the **Change Tag** tab, modify the tag parameters as required. For more information, see [Edit a factory tag](#).

View the tag task under a tag

- Go to the Tag Tasks section by using one of the following methods:
 - In the **Tag Task Instances** section, move the pointer over the  icon next to the target tag and select **View Tag Tasks**.
 - In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the Tag box and select **View Tag Tasks**.



- In the **Tag Tasks** section, view the tag task under the tag. You can view the details of the task.

View the operational logs of a tag task instance

- In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **View Operational Log**.
- On the **Operational Log** page, view the operational logs of the instance.

View the code of a tag task instance

- In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **View Node Script**.
- On the **Node Script** page, view the code of the instance.

View the operation logs of a tag task instance

1. In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **View Operations Log**.
2. In the **Operations Log** pane, view the operation logs of the instance. You can click the  icon in the **Details** column to view the operation details.

View the global DAG of the tag task for which a tag task instance was generated

1. In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **View Global Tasks**.
2. On the **Global Management** tab, view the tag task for which the instance was generated in the **Recurring Tasks** section and the global DAG of the task in the right-side workspace.

View the global DAG of a tag task instance

1. In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **View Global Task Instances**.
2. On the **Global Management** tab, view the instance in the **Recurring Task Instances** section and the global DAG of the instance in the right-side workspace.

Terminate a tag task instance

 **Note** You can terminate a tag task instance only when the instance is in the **Wait Submission** or **Running** state.

1. In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **Terminate**.
2. In **Terminate** message, click **OK**.

Rerun a tag task instance

 **Note**

- You can rerun a tag task instance only when the instance is in the **Success**, **Not Running**, or **Failed** state.
- Use caution when you rerun a tag task instance, because the data generated by the instance previously will be overwritten after the instance is rerun.

1. In the **Tag Task Instances** section, click the target tag and then the target tag task instance. In the DAG in the right-side workspace, right-click the instance node and select **Rerun and Resume Scheduling**.
2. In the **Rerun and Resume Scheduling** message, click **OK**.

9.12.4.5. Retroactive data generation instances

The **Retroactive Data Generation Instances** section displays retroactive data generation instances that are generated when you generate retroactive data for tags and behavior rules. This topic describes the operations that you can perform on retroactive data generation instances.

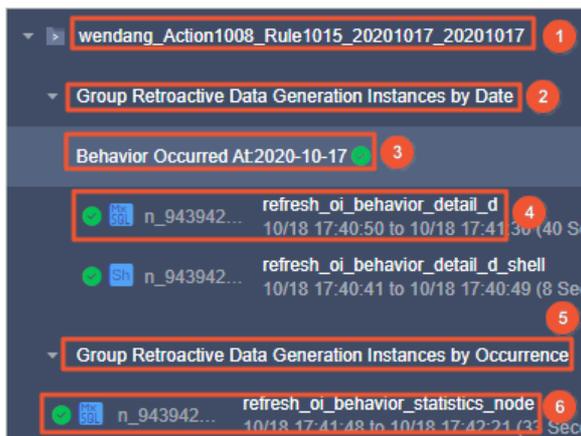
Go to the Retroactive Data Generation Instances section

1. [Log on to the Dataphin console](#).
2. Go to the **Distilling Management** tab.

- o If the project that you accessed last time is the **Data_distill** project, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner of the Dataphin homepage.
- o If the project that you accessed last time is not the **Data_distill** project, use one of the following methods to go to the **Distilling Management** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, move the pointer over **Develop** in the top navigation bar and select **Distilling**.
 - c. On the **Distilling** page, click **Scheduling** in the top navigation bar.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner. The **Scheduling** page appears.
 - b. Click the  icon next to the project name in the upper-left corner, click the **Basic** tab, and then select the **Data_distill** project.
- 3. On the left-side navigation submenu, click the  icon.
- 4. In the **Retroactive Data Generation Instances** section, view the retroactive data generation instances that are generated for tags and behavior rules.

You can also search for retroactive data generation instances in the Retroactive Data Generation Instances section. If you click a retroactive data generation instance for a behavior rule or a tag in the Retroactive Data Generation Instances section, the DAG of the instance appears in the right-side workspace. The DAG shows the node of the current retroactive data generation instance and its relationships with other retroactive data generation instances for the same behavior rule or tag. The thumbnail of the DAG appears in the lower-right corner of the workspace. The **Retroactive Data Generation Instances** section contains the **Retroactive Data Generation Instance for Behavior Rule** and **Retroactive Data Generation Instance for Tag** tabs. You can view different instance information on the two tabs.

- o The following table describes the instance information that you can view on the **Retroactive Data Generation Instance for Behavior Rule** tab.



No.	Description
1	The name of the folder for storing retroactive data generation instances for a behavior rule. The name is in the format of Prefix_Behavior ID_Behavior rule ID_Start date for generating retroactive data_End date for generating retroactive data.

No.	Description
2	<p>A type of retroactive data generation instances.</p> <ul style="list-style-type: none"> ▪ Group Retroactive Data Generation Instances by Date: This type of instances are run every day in the specified period. You can specify a date in the Group Retroactive Data Generation Instances by Date box in the DAG to filter specific instances of this type. ▪ Group Retroactive Data Generation Instances by Occurrence: This type of instances are run only once. Dataphin runs this type of instances for a behavior rule only after all instances of the Group Retroactive Data Generation Instances by Date type are properly run for the behavior rule.
3	<p>The data timestamp and overall running status of retroactive data generation instances of the Group Retroactive Data Generation Instances by Date type.</p> <ul style="list-style-type: none"> ▪ The overall running status is Success only when all the instances are properly run. ▪ The overall running status is Failed if any instance failed. ▪ The overall running status is Running if some instances are properly run and the others are running.
4	<p>The display name of a retroactive data generation instance of the Group Retroactive Data Generation Instances by Date type. The display name consists of the running status of the instance, type of the corresponding behavior rule task, ID of the instance, and name of the instance.</p>
5	<p>A type of retroactive data generation instances.</p>
6	<p>The display name of a retroactive data generation instance of the Group Retroactive Data Generation Instances by Occurrence type. The display name consists of the running status of the instance, type of the corresponding behavior rule task, ID of the instance, and name of the instance.</p>

- o The following table describes the instance information that you can view on the **Retroactive Data Generation Instance for Tag** tab.



No.	Description
1	<p>The name of the folder for storing retroactive data generation instances for a tag. The name is in the format of Prefix_Tag ID_End date for generating retroactive data.</p>

No.	Description
2	The display name of a retroactive data generation instance for a tag. The display name consists of the running status of the instance, type of the corresponding tag task, ID of the instance, and name of the instance.

Search for retroactive data generation instances

In the **Retroactive Data Generation Instances** section, retroactive data generation instances are sorted in descending order of generation time. You can search for retroactive data generation instances by using one of the following methods:

- Enter an instance name in the search box to search for specific retroactive data generation instances. On the **Retroactive Data Generation Instance for Behavior Rule** tab, the instances whose data timestamp is the current day are found by default. On the **Retroactive Data Generation Instance for Tag** tab, the instances whose data timestamp is the previous day are found by default.
- On the **Retroactive Data Generation Instance for Behavior Rule** tab, click **Today** or **Yesterday** or select a date below **Retroactive Data Generation Date** to filter retroactive data generation instances. On the **Retroactive Data Generation Instance for Tag** tab, click **Yesterday** or **2 Days Ago** or select a date below **End Date** to filter retroactive data generation instances.

View the operational logs of a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance whose operational logs you want to view. In the DAG in the right-side workspace, right-click the instance node and select **View Operational Log**.
2. On the **Operational Log** page, view the operational logs of the instance.

View the code of a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance whose code you want to view. In the DAG in the right-side workspace, right-click the instance node and select **View Node Script**.
2. On the **Node Script** page, view the code of the instance.

View the operation logs of a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance whose operation logs you want to view. In the DAG in the right-side workspace, right-click the instance node and select **View Operations Log**.
2. In the **Operations Log** pane, view the operation logs of the instance. You can click the  icon in the **Details** column to view the operation details.

Terminate a retroactive data generation instance

 **Note** You can terminate a retroactive data generation instance only when the instance is in the **Wait Submission** or **Running** state.

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance that you want to terminate. In the DAG in the right-side workspace, right-click the instance node and select **Terminate**.
2. In **Terminate** message, click **OK**.

Rerun a retroactive data generation instance

 **Note**

- You can rerun a retroactive data generation instance only when the instance is in the **Success**, **Not Running**, or **Failed** state.
- Use caution when you rerun a retroactive data generation instance, because the data generated by the instance previously will be overwritten after the instance is rerun.

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance that you want to rerun. In the DAG in the right-side workspace, right-click the instance node and select **Rerun and Resume Scheduling**.
2. In the **Rerun and Resume Scheduling** message, click **OK**.

Rerun the downstream nodes of a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance whose downstream nodes you want to rerun. In the DAG in the right-side workspace, right-click the instance node and select **Rerun Downstream Nodes**.
2. In the **Rerun Downstream Nodes** dialog box, select the nodes to be rerun and click **OK**.

Set the running status of a retroactive data generation instance to Success and schedule the instance

 **Note** You can set the running status of a retroactive data generation instance to **Success** and schedule the instance only when the instance is in the **Failed** state.

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance for which you want to set the running status to **Success** for scheduling. In the DAG in the right-side workspace, right-click the instance node and select **Set to Succeeded and Resume Scheduling**.
2. In the **Set to Succeeded and Resume Scheduling** message, click **OK**.

Forcibly rerun a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance that you want to forcibly rerun. In the DAG in the right-side workspace, right-click the instance node and select **Force Rerun**.
2. In the **Force Rerun** message, click **OK**.

 **Note** Use caution when you rerun a retroactive data generation instance, because the data generated by the instance previously will be overwritten after the instance is rerun.

Pause a retroactive data generation instance

1. In the **Retroactive Data Generation Instances** section, click the retroactive data generation instance that you want to pause. In the DAG in the right-side workspace, right-click the instance node and select **Pause**.
2. In the **Pause** message, click **OK**.

9.13. Monitoring and alerting

9.13.1. Alert records for batch processing tasks

In Dataphin, you can configure alert rules for batch processing tasks. When the alert rules are triggered, Dataphin sends alert notifications and generates alert records. This topic describes how to view the alert records and alert details of batch processing tasks.

Go to the Batch Processing Alert Records page

1. [Log on to the Dataphin console.](#)
2. Go to the **Batch Processing Alert Records** page.
 - If the project that you accessed last time is in **Prod** or **Basic** mode, use one of the following methods to go to the **Batch Processing Alert Records** page:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane. The **Batch Processing Alert Records** page appears.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane. The **Batch Processing Alert Records** page appears.
 - If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Batch Processing Alert Records** page:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - d. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane. The **Batch Processing Alert Records** page appears.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane. The **Batch Processing Alert Records** page appears.

On the **Batch Processing Alert Records** page, you can filter alert records of batch processing tasks by alert cause, notification method, notification status, recipient, and notification time. You can also search for alert records by entering a keyword of task names in the search box. In the lower part of the **Batch Processing Alert Records** page, you can view alert records of batch processing tasks in detail, including the task name and ID, monitoring type, alert cause, notification method, notification status, recipient, notification time, and actions that you can perform on each alert record.

Parameter	Description
Task Name and ID	The name and ID of the batch processing task that triggered the alert.
Monitoring Type	The type of the alert.

Parameter	Description
Alert Cause	<p>The cause of the alert. Valid values:</p> <ul style="list-style-type: none"> ◦ Complete ◦ Error ◦ Timeout ◦ Incomplete
Notification Method	<p>The method to notify the recipients of the alert. Valid values:</p> <ul style="list-style-type: none"> ◦ Email ◦ SMS ◦ Cellphone ◦ DingTalk Group Chatbot
Notification Status	<p>The status of the alert notification. Valid values:</p> <ul style="list-style-type: none"> ◦ Success: The system has sent the alert notification by using the specified notification method, and received a receipt. ◦ Failed: One of the following errors occurred: <ul style="list-style-type: none"> ▪ The system failed to send the alert notification by using the specified notification method due to one of the following reasons: <ul style="list-style-type: none"> ▪ The mobile number, email address, or webhook URL of the recipient was not specified. ▪ The number of SMS messages, emails, or DingTalk messages that the recipient can receive on the current day reached the upper limit. ▪ The number of SMS messages, emails, or DingTalk messages that the system can send on the current day reached the upper limit. ▪ The recipient was not in the Dataphin member list. ▪ The system sent the alert notification by using the specified notification method but received an error response. For example, the system received an error response indicating that the mobile number was invalid or the email address did not exist. ▪ A system error occurred. You can view the cause of the failure to obtain the detailed error message. ◦ Sending: The system has submitted a task to send the alert notification, and the task is queuing.
Recipient	The users to whom the alert notification was sent.
Alerted At	The time when the alert notification was sent to the recipients.
Actions	The actions that you can perform on the alert record.

View alert details

1. On the **Batch Processing Alert Records** page, find the alert record whose details you want to view and click the  icon in the **Actions** column.
2. In the **Alert Details** message, you can view the task ID, triggering time, alert content, and alert rule

creator.

9.13.2. Alert rules for batch processing tasks

In Dataphin, you can configure alert rules to monitor the running status of batch processing tasks in the production environment. This topic describes how to create, modify, delete, enable, and disable alert rules for batch processing tasks.

Go to the Batch Processing Monitor Settings page

1. [Log on to the Dataphin console.](#)
2. Go to the **Monitoring** tab.
 - If the project that you accessed last time is in **Prod** or **Basic** mode, use one of the following methods to go to the **Monitoring** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Monitoring** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - d. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
3. On the **Monitoring** tab, click the  icon on the left-side navigation submenu. The **Batch Processing Monitor Settings** page appears.

On the **Batch Processing Monitor Settings** page, you can filter alert rules for batch processing tasks by alert cause, notification method, recipient, and alert rule creator. You can also search for alert rules by entering a keyword of task names in the search box. In the lower part of the **Batch Processing Monitor Settings** page, you can view alert rules for batch processing tasks in detail, including the task name and ID, alert cause, notification method, recipient, alert rule creator, monitoring status, and actions that you can perform on each alert rule.

Parameter	Description
Task Name and ID	The name and ID of the batch processing task that is monitored based on the alert rule.

Parameter	Description
Alert Cause	The cause of alerts. Valid values: <ul style="list-style-type: none"> ◦ Complete ◦ Error ◦ Timeout ◦ Incomplete
Notification Method	The method to notify the alert recipients. Valid values: <ul style="list-style-type: none"> ◦ Email ◦ SMS ◦ Cellphone ◦ DingTalk Group Chatbot
Recipient	The users to whom the alert notifications will be sent.
Created By	The user who created the alert rule.
Monitor Switch	Indicates whether the alert rule is enabled for the task. <ul style="list-style-type: none"> ◦ : The alert rule is enabled for the task. ◦ : The alert rule is disabled for the task. You can click the switch to enable or disable the alert rule for the task.
Actions	The actions that you can perform on the alert rule. <ul style="list-style-type: none"> ◦ You can click the  icon to modify the alert rule. ◦ You can click the  icon to delete the alert rule.

Create an alert rule for a batch processing task

1. On the **Batch Processing Monitor Settings** page, click **Create Batch Processing Task Monitor** in the upper-right corner.
2. In the **Create Batch Processing Task Monitor** dialog box, set the parameters as required.

Parameter	Description
Select Task	The batch processing task to be monitored. You can select a maximum of three tasks.
Alert Cause	The cause of alerts. Valid values: <ul style="list-style-type: none"> ◦ Complete ◦ Error ◦ Timeout. If you select this option, you must specify a timeout period. ◦ Incomplete. If you select this option, you must specify a time point.

Parameter	Description
Recipient	<p>The users to whom the alert notifications will be sent. Valid values:</p> <ul style="list-style-type: none"> ◦ Owner: the owner of the batch processing task. ◦ Custom Recipients: the specified users to whom the alert notifications will be sent. You can specify a maximum of three alert recipients for a batch processing task.
Notification Method	<p>The method to notify the alert recipients. Valid values:</p> <ul style="list-style-type: none"> ◦ Email ◦ SMS ◦ Cellphone ◦ DingTalk Group Chatbot

 **Note**

- For an Alibaba Cloud account, the system can send a maximum of 100 messages by SMS, 2,000 messages by email, and 2,000 messages by DingTalk chatbot per day.
- An Alibaba Cloud account can receive a maximum of 20 messages by SMS, 400 messages by email, and 400 messages by DingTalk chatbot per day.
- For an Alibaba Cloud account, the system can send a maximum of 100 voice messages per day. A voice message that is shorter than 1 minute is billed as 1 minute.

3. Click OK.

Modify an alert rule for a batch processing task

1. On the **Batch Processing Monitor Settings** page, find the alert rule that you want to modify and click the  icon in the **Actions** column.
2. In the **Change Task Monitoring** dialog box, modify the parameters as required. For more information, see [Create an alert rule for a batch processing task](#).
3. Click OK.

Delete an alert rule from a batch processing task

1. On the **Batch Processing Monitor Settings** page, find the alert rule that you want to delete and click the  icon in the **Actions** column.
2. In the Delete message, click OK.

Enable or disable an alert rule for a batch processing task

On the **Batch Processing Monitor Settings** page, find the alert rule that you want to enable or disable and click the  or  icon in the **Monitor Switch** column.

9.13.3. Alert records for stream processing tasks

In Dataphin, you can configure alert rules for stream processing tasks. When the alert rules are triggered, Dataphin sends alert notifications and generates alert records. This topic describes how to view the alert records and alert details of stream processing tasks.

1. [Log on to the Dataphin console.](#)
2. Go to the **Monitoring** tab.
 - If the project that you accessed last time is in **Prod** or **Basic** mode, use one of the following methods to go to the **Monitoring** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
- If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Monitoring** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - d. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
- a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
- b. On the **Scheduling** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
- c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
3. On the **Monitoring** tab, click the  icon on the left-side navigation submenu. The **Stream Processing Alert Records** page appears.

On the **Stream Processing Alert Records** page, you can filter alert records of stream processing tasks by alert cause, notification status, recipient, and notification time. You can also search for alert records by entering a keyword of task names in the search box. In the lower part of the **Stream Processing Alert Records** page, you can view alert records of stream processing tasks in detail, including the object name, last triggering time, recipient, alert cause, and notification status of each alert record.

Parameter	Description
Object Name	The name of the stream processing task that triggered the alert.
Last Alerted At	The time when the latest alert was triggered.

Parameter	Description
Recipient	The users to whom the alert notification was sent. Valid values: <ul style="list-style-type: none"> ◦ Owner: the owner of the stream processing task. ◦ Custom: the specified users to whom the alert notification was sent. A stream processing task supports a maximum of five alert recipients. ◦ Shift Schedule: the shift schedule of users to whom the alert notification was sent. A stream processing task supports a maximum of one shift schedule.
Alert Cause	The cause of the alert. Valid values: <ul style="list-style-type: none"> ◦ Excessively Long Delay ◦ Exceeds Maximum TPS ◦ Exceeds Specified Failure Frequency ◦ Exceeds Specified Data Retention Period
Status	The status of the alert notification. Valid values: <ul style="list-style-type: none"> ◦ Alerting ◦ Do-Not-Disturb ◦ Completed ◦ Alerting (Silence Period)

9.13.4. Alert rules for stream processing tasks

In Dataphin, you can configure alert rules to monitor the running status of stream processing tasks in the production environment. This topic describes how to create, modify, delete, enable, and disable alert rules for stream processing tasks.

Go to the Stream Processing Monitor Settings page

1. [Log on to the Dataphin console.](#)
2. Go to the **Monitoring** tab.
 - If the project that you accessed last time is in **Prod** or **Basic** mode, use one of the following methods to go to the **Monitoring** tab:
 - a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
- If the project that you accessed last time is in **Dev** mode, use one of the following methods to go to the **Monitoring** page:

- a. On the Dataphin homepage, click **R&D** in the top navigation bar.
 - b. On the **Develop** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Develop** page, click **Scheduling** in the top navigation bar.
 - d. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
 - a. On the Dataphin homepage, click **Scheduling and Management** in the middle section, or click **Enter** next to **Scheduling** in the upper-right corner.
 - b. On the **Scheduling** page, click the  icon next to the project name in the upper-left corner and select a project in **Prod** or **Basic** mode.
 - c. On the **Scheduling** page, click the **Monitoring** tab in the left-side navigation pane.
3. On the **Monitoring** tab, click the  icon on the left-side navigation submenu. The **Stream Processing Monitor Settings** page appears.

On the **Stream Processing Monitor Settings** page, you can filter alert rules for stream processing tasks by alert cause, notification method, recipient, and alert rule creator. You can also search for alert rules by entering a keyword of task names in the search box. In the lower part of the **Stream Processing Monitor Settings** page, you can view alert rules for stream processing tasks in detail, including the task name and ID, alert cause, notification method, recipient, alert rule creator, monitoring status, and actions that you can perform on each alert rule.

Parameter	Description
Task Name and ID	The name and ID of the stream processing task that is monitored based on the alert rule.
Alert Cause	The cause of alerts. Valid values: <ul style="list-style-type: none"> ○ Excessively Long Delay ○ Exceeds Maximum TPS ○ Exceeds Specified Failure Frequency ○ Exceeds Specified Data Retention Period
Notification Method	The method to notify the alert recipients. Valid values: <ul style="list-style-type: none"> ○ Email ○ SMS ○ Cellphone ○ DingTalk Group Chatbot
Recipient	The users to whom the alert notifications will be sent.
Created By	The user who created the alert rule.
Last Modified At	The time when the alert rule was last modified.

Parameter	Description
Monitor Switch	<p>Indicates whether the alert rule is enabled for the task.</p> <ul style="list-style-type: none"> : The alert rule is enabled for the task. : The alert rule is disabled for the task. <p>You can click the switch to enable or disable the alert rule for the task. You can also click the  icon next to the switch to delete the alert rule.</p>
Actions	<p>The actions that you can perform on the alert rule. Valid values:</p> <ul style="list-style-type: none"> You can click the  icon to modify the alert rule. You can click the  icon to delete the alert rule.

Create an alert rule for a stream processing task

- On the **Stream Processing Monitor Settings** page, click **Create Stream Processing Task Monitor** in the upper-right corner.
- In the **Create Stream Processing Task Monitor** dialog box, set the parameters as required.

Section	Parameter	Description
Select Task	Task	The stream processing task to be monitored.
Specify Alert Causes	Alert Causes	<p>The alert cause of the alert rule. Click the  icon next to Alert Causes to add an alert rule with another alert cause. You can configure the following alert rules for a stream processing task:</p> <ul style="list-style-type: none"> Exceeds Specified Data Retention Period: You must specify an upper limit for the queued time, in seconds. When the queued time exceeds this limit, an alert is triggered. Excessively Long Delay: You must specify an upper limit for the processing delay, in seconds. When the processing delay exceeds this limit, an alert is triggered. Exceeds Maximum TPS: You must specify a lower limit and an upper limit for the TPS. When the TPS is beyond the specified range, an alert is triggered. Exceeds Specified Failure Frequency: You must specify an upper limit for the number of failures that occur in a minute. If the number of failures that occur in a minute exceeds this limit, an alert is triggered. <p>You can click the  icon next to an alert rule to delete the alert rule.</p>
	Alerted Every	The time interval between alerts, in minutes. Valid values: 1 to 59.

Section	Parameter	Description
	Time Range	The period during which the task is monitored. Valid values: <ul style="list-style-type: none"> ◦ All Day ◦ Specified
	Recipients	The users to whom the alert notifications will be sent. Valid values: <ul style="list-style-type: none"> ◦ Owner: the owner of the stream processing task. ◦ Custom: the specified users to whom the alert notifications will be sent. You can specify a maximum of five alert recipients for a stream processing task. ◦ Shift Schedule: the shift schedule of users to whom the alert notifications will be sent. You can specify a maximum of one shift schedule.
	Notification Method	The method to notify the alert recipients. Valid values: <ul style="list-style-type: none"> ◦ Email ◦ SMS ◦ Cellphone ◦ DingTalk

 **Note**

- For an Alibaba Cloud account, the system can send a maximum of 100 messages by SMS, 2,000 messages by email, and 2,000 messages by DingTalk chat bot per day.
- An Alibaba Cloud account can receive a maximum of 20 messages by SMS, 400 messages by email, and 400 messages by DingTalk chat bot per day.
- For an Alibaba Cloud account, the system can send a maximum of 100 voice messages per day. A voice message that is shorter than 1 minute is billed as 1 minute.

3. Click **OK**.

Modify an alert rule for a stream processing task

1. On the **Stream Processing Monitor Settings** page, find the alert rule that you want to modify and click the  icon in the **Actions** column.
2. In the **Edit Stream Processing Task Monitor** dialog box, modify the parameters as required. For more information, see [Create an alert rule for a stream processing task](#).
3. Click **OK**.

Delete alert rules for a stream processing task

1. On the **Stream Processing Monitor Settings** page, find the alert rules that you want to delete and click the  icon in the **Actions** column.
2. In the **Tip** message, click **OK**.

Enable or disable an alert rule for a stream processing task

On the **Stream Processing Monitor Settings** page, find the alert rule that you want to enable or disable and click the  or  icon in the **Monitor Switch** column.

Delete an alert rule for a stream processing task

1. On the **Stream Processing Monitor Settings** page, find the alert rule that you want to delete and click the  icon in the **Monitor Switch** column.
2. In the **Tip** message, click **OK**.

9.14. Data assets

9.14.1. Overview

This topic describes the Data Assets module of Dataphin and the three modes of viewing data assets, that is, the Global, Flow, and Structure modes.

Data asset overview

After development works such as data collection, integration, and processing are completed, you can manage data assets in a systematic way.

Based on the standards and methodology of data asset management, the Data Assets module of Dataphin allows you to take inventory of and evaluate the data assets in your enterprise, including:

- Automatically extracts and analyzes metadata, and creates a data asset dashboard. This helps enterprise managers discover and understand the value of data assets.
- Provides an end-to-end inventory check and analysis of computing, storage, security, and applications during data production. This helps you detect problems, propose and implement governance optimization schemes, reduce costs, and improve efficiency for data.
- Provides the Data Assets page for you to view the data modeling results and table details.

You can view data assets in **Global**, **Flow**, and **Structure** modes.

- The **Global** mode displays business units that contain a large amount of data in the form of planets and their respective data sizes.
- The **Flow** mode displays the entire process of data ingestion, integration, and output. This reveals the underlying potential capabilities of a Data Mid-End.
- The **Structure** mode displays components in different shapes to represent business entities and uses lines of different styles to represent relationships between these entities. This mode clearly shows the structure of data in a business unit.

Global mode

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the Data Assets page, click **Overview** in the top navigation bar.
4. On the **Overview** page, click **Global** in the upper-right corner.
5. The **Global** tab displays the total number of tasks, tables, and projects in Dataphin in the lower-right corner. It also displays the rankings of business units that contain a large amount of data, including the business unit name, data size, and proportion to the total data size. You can also move the pointer over a planet to view the statistics about the business unit, including the amount of used computing resources and storage size.

 **Note** The **Overview** page only displays information about the metadata of the production environment and the application data that has been processed.

Flow mode

On the **Overview** page, click **Flow** in the upper-right corner.

The **Flow** tab displays the data ingestion progress, the total number of tables, and data applications such as data query in a visual view.

- Move the pointer over **Data Ingestion Progress** to view the number of data sources.
- Move the pointer over **Total Tables** to view the statistics about tables from the perspectives of data standardization and data models.

Structure mode

On the **Overview** page, click **Structure** in the upper-right corner. On the **Structure** tab, click a business unit in the upper-right corner to view all the dimensions, business processes, and their relationships under this business unit. The **Structure** tab displays the following information in the upper-left corner:

- **Dimensions:** the types of dimensions. Dimensions are classified into common dimensions and other dimensions. Other dimensions include common dimensions by hierarchy, enumeration dimensions, and virtual dimensions.
- **Relation:** the relationships between dimensions.
 -  indicates a parent-child relationship.
 -  indicates an association relationship.
- Search box: You can enter a dimension or business process name in the **Search** field or click the  icon next to **Search**, and select a dimension or business process from the drop-down list. The dimensions and business processes related to the selected dimension or business process appear.

You can use one of the following methods to view relationships between dimensions and between dimensions and business processes:

- Click a dimension. The dimensions and business processes related to this dimension are highlighted.
- Click a business process. The dimensions related to this business process are highlighted.
- Use the search box in the upper-left corner of the **Structure** tab to search for the desired dimension or business process and view its related dimensions and business processes.

9.14.2. Map

9.14.2.1. View the asset map

An asset map summarizes the relationships between dimensions and business processes in a data domain of a business unit to show the structure of your enterprise data. In addition, the asset map allows you to efficiently and accurately search for and explore data based on features such as data search, access recording, and bookmarking.

Map page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar. The **Map** page appears by default.

The Map page consists of the following four sections:

- Search bar
- Business Units section
- My Data Assets section
- Recommended section

Search bar



- You can enter a keyword in the search box to search for tables whose names contain the keyword in the development or production environment.
- By default, when you enter a keyword in the search box, 10 related tables and multiple related keywords that match the keyword appear in the drop-down list. You can view the names, display names, and descriptions of the related tables in the drop-down list.
- If you select a related table, the table details page appears. If you select a related keyword, the search results are updated based on the selected keyword.

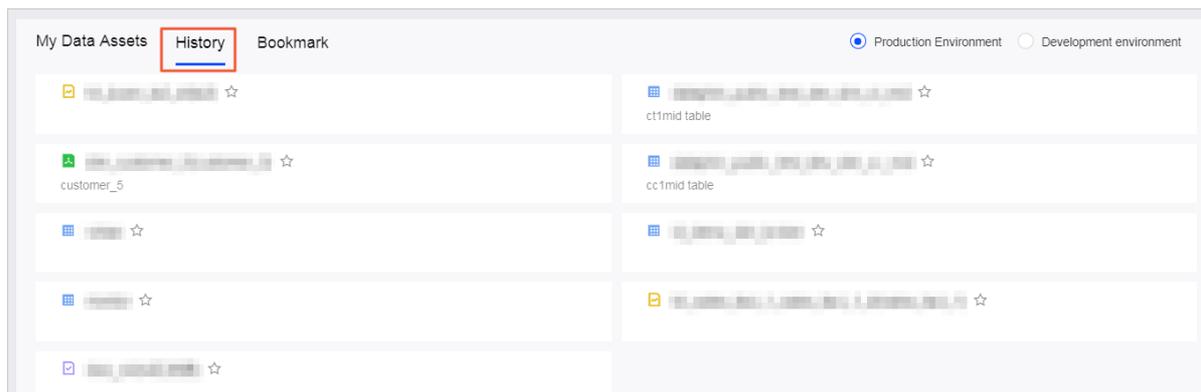
Business Units section

The **Business Units** section is below the search bar. In this section, the division of information echoes the asset overview. This section only displays the data of the production environment, including the business units, data domains, dimensions, and business processes.

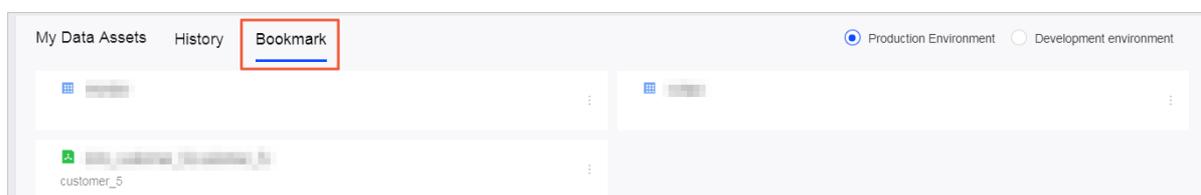
My Data Assets section

This section displays the data assets that you have accessed or bookmarked. You can switch between the development environment and production environment to view the access history and bookmarks. By default, the My Data Assets section displays the access history and bookmarks of the production environment.

- The **History** tab displays the 10 data assets that you recently accessed.

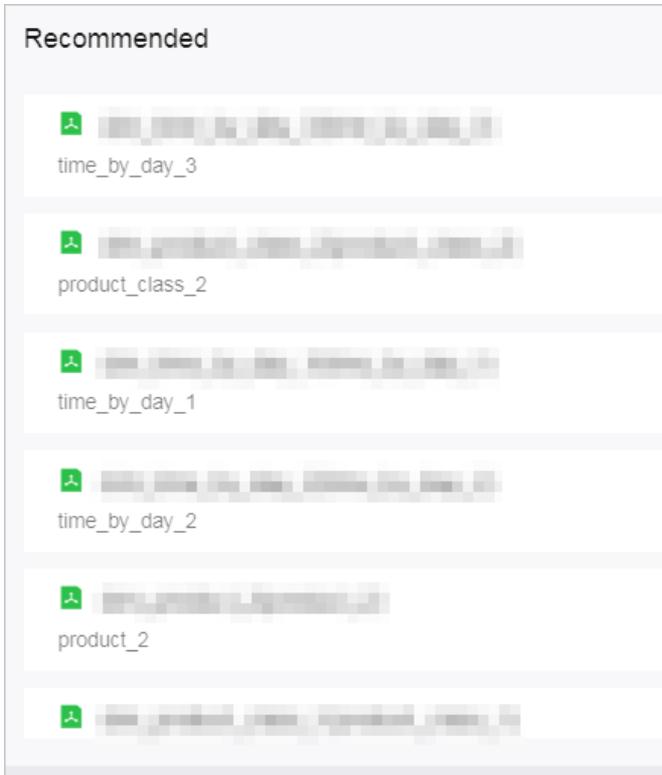


- The **Bookmark** tab displays the data assets that you bookmarked on the asset details page. The number of data assets that can be bookmarked is not limited.



Recommended section

This section displays the most popular tables, including their names, display names, and descriptions, based on the popularity of global user access.



Metadata

No search results will be returned if you query a table that is not created in Dataphin or a newly created table whose metadata has not been synchronized because of the latency of Dataphin in obtaining metadata. In this case, you can manually synchronize the metadata of the table to Dataphin.

1. On the **Map** page, click **Search** in the search bar.
2. On the page that appears, click the  icon in the upper-right corner. The **Refresh Metadata** dialog box appears.
3. Set **Project Name** and **Table Name** and click **OK**.
 - **Project Name**: the name of the project to which the table belongs. You can view all projects of the current tenant in the drop-down list. Select one from the list.
 - **Table Name**: the name of the table whose metadata needs to be synchronized to Dataphin. You must enter an exact match to the table name.

 **Note** Make sure that the table name is correct before you click **OK**.

9.14.2.2. View table details

This topic describes how to search for physical tables, logical tables, and stream metatables, and view table details.

Search for tables

1. Log on to the Dataphin console.
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar. The **Map** page appears by default.
3. Enter a keyword in the **search box** and press Enter or click **Search**. Alternatively, select a dimension or business process in the **Business Units** section. The Search Results page appears.

The Search Results page provides the following information and features:

- The left-side pane displays filter conditions and the right-side pane displays the tables that match the selected filter conditions. Filter conditions fall into the following two categories:
 - **Business Properties:** Filter conditions in this category are further classified by project, business unit, data domain, dimension, and business process.
 - **Data Properties:** Filter conditions in this category are further classified by environment, storage type, and table type, such as logical dimension table, logical fact table, and logical aggregate table.
- You can select one or more filter conditions in the left-side pane to filter tables and update the search results in the right-side pane. By default, Dataphin displays only one row of filter conditions for each type. You can click the downwards arrow to view more filter conditions. You can click the name of a table in the right-side pane to go to the details page of the table.

View the details about a physical table

Note The table details page varies with the table type.

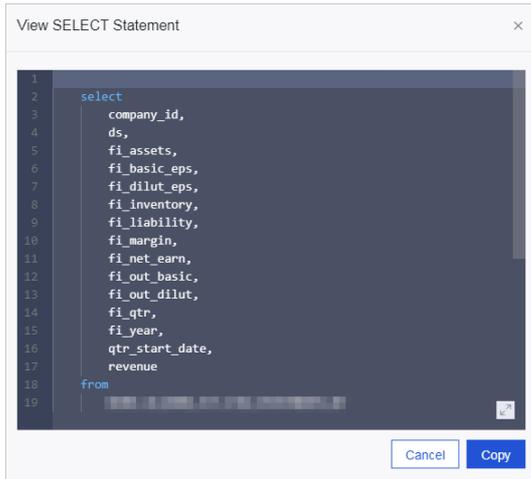
On the Search Results page, click the name of a physical table. The details page of the table appears, as shown in the following figure.

Field Name	Field Display Name	Data Type	Popularity
1	id_type	id_type	STRING
2	id_value	id_value	STRING
3	prefer	prefer	STRING
4	behavior_count	behavior_count	BIGINT
5	confidence	confidence	DOUBLE

The following field are partitioning fields.			
1	ds	date partition	STRING

- The upper-left part of the details page provides the following information and features:
 - **Table type:** The type of a table can be Logical Dimension Table, Logical Fact Table, Logical Aggregate Table, Physical Table, or Stream Metatable. In this example, the table type is **Physical Table**.
 - **Table name:** The table name is in the format of Business unit name. Table name. A maximum of 70 characters can be displayed for the name. The excess part is displayed as an ellipsis (...).
 - **Apply for Permissions:** Click Apply for Permissions. On the Apply for Physical Table Permissions tab, set the parameters to apply for required permissions.

- **Bookmark or Remove Bookmark:** The action varies depending on whether the table is bookmarked.
 - The ☆ icon indicates that you have not bookmarked the table. You can click the ☆ icon to bookmark the table.
 - The ★ icon indicates that you have bookmarked the table. You can click the ★ icon to remove the table from your bookmarks.
- **Export:** You can click **Export** to export the fields in the table to a CSV file.
- **View SELECT Statement:** Click **View SELECT Statement**. In the **View SELECT Statement** dialog box, you can view the SQL statement for querying the table data.



- In the right-side pane of the details page, you can view the basic information, access information, physical information, and update information about the table.

The lower-left part of the details page provides the following tabs:

- **Table Structure:** This tab displays the name, display name, data type, and popularity of the fields in the physical table.
- **Partition Overview:** This tab displays the partition information about the physical table.
- **Data Preview:** This tab displays the table content, including the name and records of each metric.

Note If you are not authorized to view the records, each record is de-identified with multiple asterisks (*****).

- **Tasks:** This tab displays the scheduling information about the tasks that are related to the physical table, including the ID, execution time in seconds, start time, and end time of each task.
 - You can click the 📅 icon in the **Task ID** column of a task to go to the **Recurring Task Instances** page. On this page, you can view the instances that have been generated for the task.
 - You can click the 📄 icon in the **Task ID** column of a task to go to the **Node Script** page. On this page, you can view the code of the task.
- **Data Lineage:** This tab displays the dependencies between tables.

View the details about a logical table

Note This section describes how to view the details about a logical fact table. Follow the same procedure for other types of logical tables.

On the Search Results page, click the name of a logical fact table. The details page of the table appears, as shown in the following figure.

Field Name	Field Display Name	Data Type	Popularity	Actions
1	r_age	STRING		
2	r_town	STRING		
3	r_name	STRING		
4	r_work	STRING		
5	r_salary	STRING		
6	id	STRING		

The following field are partitioning fields.

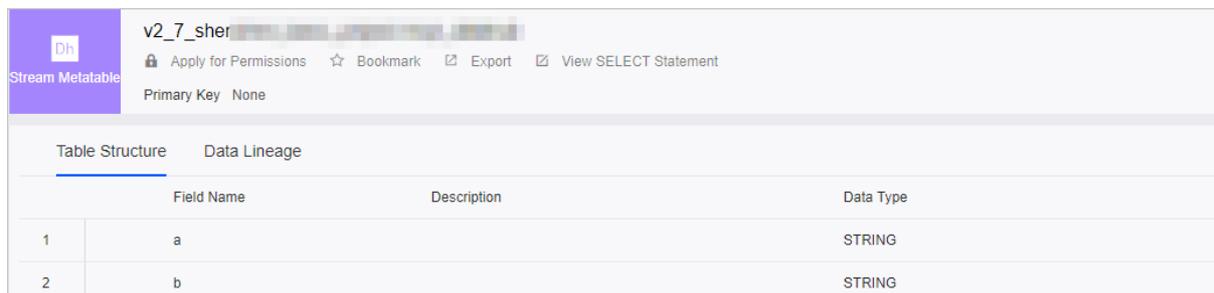
1	ds	STRING		
---	----	--------	--	--

The details page consists of the upper-left part, lower-left part, and right-side pane.

- In the upper-left part, you can view the type, name, primary table, primary key, and filter conditions of the logical fact table. You can also perform the following actions on the logical fact table: **Apply for Permissions**, **Bookmark** or **Remove Bookmark**, **Export**, and **View SELECT Statement**.
- In the right-side pane, you can view the basic information, access information, physical information, and update information about the logical fact table.
- The **Table Structure** tab in the lower-left part displays the name, display name, data type, and popularity of the fields in the logical fact table. On this tab, you can perform the following operations:
 - Click the icon in the **Actions** column of a field. In the dialog box that appears, you can view the partitions, records, creation time, and update time of the field.
 - Click the icon in the **Actions** column of a field. In the dialog box that appears, you can view the numerical distribution and numerical interval distribution of the field, as well as the data records.
 - Click the icon in the **Actions** column of a field. In the dialog box that appears, you can view the data lineage of the field.
- The **Partition Overview** tab in the lower-left part displays the distribution of partitions for each field in the logical fact table. You can filter data by field or partition.
- The **Tasks** tab in the lower-left part displays the scheduling information about the tasks that are related to the logical fact table, including the ID, execution time in seconds, start time, and end time of each task. On this tab, you can perform the following operations:
 - Click the icon in the **Task ID** column of a task to go to the **Recurring Task Instances** page. On this page, you can view the instances that have been generated for the task.
 - Click the icon in the **Task ID** column of a task to go to the **Node Script** page. On this page, you can view the code of the task.

View the details about a stream metatable

On the Search Results page, click the name of a stream metatable. The details page of the table appears, as shown in the following figure.



	Field Name	Description	Data Type
1	a		STRING
2	b		STRING

The details page consists of the upper-left part, lower-left part, and right-side pane.

- By default, the upper-left part displays the type, name, and primary key of the stream metatable.
- In the right-side pane, you can view the basic information, access information, physical information, and update information about the stream metatable.
- The **Data Lineage** tab in the lower-left part displays the data lineage of the stream metatable.
- The **Table Structure** tab in the lower-left part displays the fields in the stream metatable.

If you have the query permission on the table, you can view the first 10 fields in the table. If you do not have the query permission on the table, click **Apply for Permissions** to apply for permissions on the table. For more information, see [Manage permissions on stream metatables](#).

9.14.3. Security

9.14.3.1. My permissions

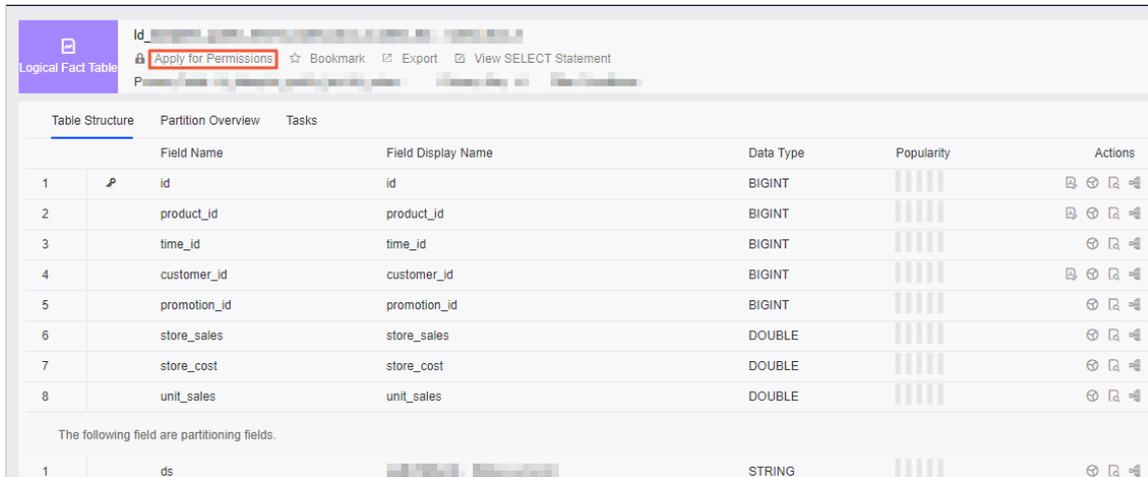
9.14.3.1.1. Manage permissions on logical tables

This topic describes how to apply for, query, renew, and remove permissions on logical tables.

Apply for permissions on a logical table

1. [Log on to the Dataphin console](#).
2. Go to the **Logical Table Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Logical Table Permissions** under **My Permissions** in the left-side navigation pane.
3. On the **Logical Table Permissions** tab, go to the **Apply for Logical Table Permissions** tab by using one of the following methods:
 - Click **Apply for Logical Table Permissions** in the upper-right corner.
 - In the left-side navigation pane, move the pointer over **Logical Table Permissions** and click the  icon.
 - In the left-side navigation pane, click the  icon next to **My Permissions** and select **Logical Table Permissions**.
 - a. In the top navigation bar, click **Map**.

- b. On the **Map** page, enter a keyword in the search box to search for tables whose names contain the keyword.
- c. Find the logical table on which you want to apply for permissions and click the table name. On the details page that appears, click **Apply for Permissions** in the upper-left part.



4. On the **Apply for Logical Table Permissions** tab, perform the following operations in the configuration wizard:
 - i. In the **Select Data Objects** step, set the **Environment**, **Business Unit**, and **Logical Table** parameters.
 - ii. Click **Next**.
 - iii. In the **Apply for Permissions** step, set the parameters as required.

Parameter	Description
Authorized Fields	The fields on which you want to apply for permissions. Select fields by using one of the following methods: <ul style="list-style-type: none"> ■ Enter a keyword in the search box to search for the desired fields and select the fields. ■ Select fields from the field list of the logical table.
Permission Type	The permission that you want to apply for. By default, the value Query is selected and you cannot modify the setting.
Account Type	The accounts to be granted the permission that you are applying for. You can grant the permission to either or both of the personal account and project production account. <ul style="list-style-type: none"> ■ The personal account is used for data modeling and development in the development environment and publishing developed objects or tasks to the production environment. If you select Personal Account, you must also set the Expiration Date parameter. ■ The project production account is used to manage and run tasks that are submitted to the production environment. If you select Project Production Account, you must also set the Project parameter.
Reason for Application	The reason why you apply for the permission.

5. Click **Submit**.

Query the permissions on a logical table

1. On the **Logical Table Permissions** tab, find the logical table on which you want to query the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, view the fields on which you have applied for permissions and the specific permissions.

Renew the permissions on a logical table

1. On the **Logical Table Permissions** tab, find the logical table on which you want to renew the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, select the fields on which you want to renew permissions.
 - o Enter a keyword in the search box to search for the desired fields and select the fields.
 - o Select fields from the field list of the logical table.
3. In the lower part of the page, click **Extend Validity**.
4. In the **Apply for Permissions** step, set the parameters as required. For more information, see [Apply for permissions on a logical table](#).
5. Click **Submit**.

Remove the permissions on a logical table

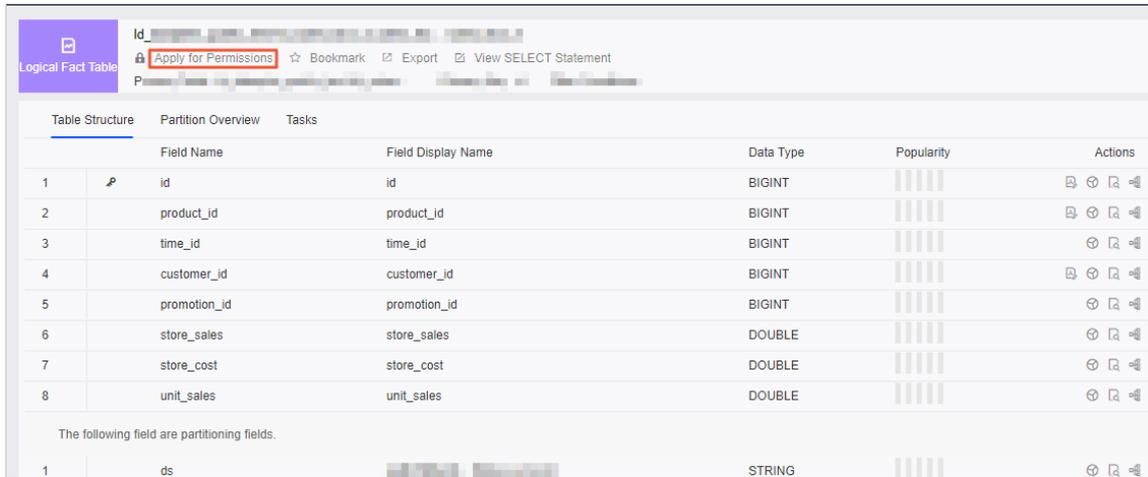
1. On the **Logical Table Permissions** tab, find the logical table on which you want to remove the permissions and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.2. Manage permissions on physical tables

This topic describes how to apply for, query, renew, and remove permissions on physical tables.

Apply for permissions on a physical table

1. [Log on to the Dataphin console](#).
2. Go to the **Physical Table Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Physical Table Permissions** under **My Permissions** in the left-side navigation pane.
3. On the **Physical Table Permissions** tab, go to the **Apply for Physical Table Permissions** tab by using one of the following methods:
 - o Click **Apply for Physical Table Permissions** in the upper-right corner.
 - o In the left-side navigation pane, move the pointer over **Physical Table Permissions** and click the  icon.
 - o In the left-side navigation pane, click the **Add** icon next to **My Permissions** and select **Physical Table Permissions**.
 - o



- a. In the top navigation bar, click **Map**.
 - b. On the **Map** page, enter a keyword in the search box to search for tables whose names contain the keyword.
 - c. Find the physical table on which you want to apply for permissions and click the table name. On the details page that appears, click **Apply for Permissions** in the upper-left part.
4. On the **Apply for Physical Table Permissions** tab, perform the following operations in the configuration wizard:
- i. In the **Select Data Objects** step, set the **Environment**, **Project**, and **Physical Tables** parameters.
 - ii. Click **Next**.
 - iii. In the **Apply for Permissions** step, set the parameters as required.

Parameter	Description
Authorized Fields	<p>The fields on which you want to apply for permissions. Select fields by using one of the following methods:</p> <ul style="list-style-type: none"> ▪ Enter a keyword in the search box to search for the desired fields and select the fields. ▪ Select fields from the field list of the physical table.
Permission Type	<p>The permissions that you want to apply for. Valid values:</p> <ul style="list-style-type: none"> ▪ Query ▪ Write ▪ Update ▪ Delete
Account Type	<p>The accounts to be granted the permissions that you are applying for. You can grant the permissions to either or both of the personal account and project production account.</p> <ul style="list-style-type: none"> ▪ The personal account is used for data modeling and development in the development environment and publishing developed objects or tasks to the production environment. ▪ The project production account is used to manage and run tasks that are submitted to the production environment.
Reason for Application	The reason why you apply for the permissions.

5. Click **Submit**.

Query the permissions on a physical table

1. On the **Physical Table Permissions** tab, find the physical table on which you want to query the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, view the fields on which you have applied for permissions and the specific permissions.

Renew the permissions on a physical table

1. On the **Physical Table Permissions** tab, find the physical table on which you want to renew the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, select the fields on which you want to renew permissions.
 - Enter a keyword in the search box to search for the desired fields and select the fields.
 - Select fields from the field list of the physical table.
3. In the lower part of the page, click **Extend Validity**.
4. In the **Apply for Permissions** step, set the parameters as required. For more information, see [Apply for permissions on a physical table](#).
5. Click **Submit**.

Remove the permissions on a physical table

1. On the **Physical Table Permissions** tab, find the physical table on which you want to remove the permissions and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.3. Manage permissions on stream metatables

This topic describes how to apply for, query, renew, and remove permissions on stream metatables.

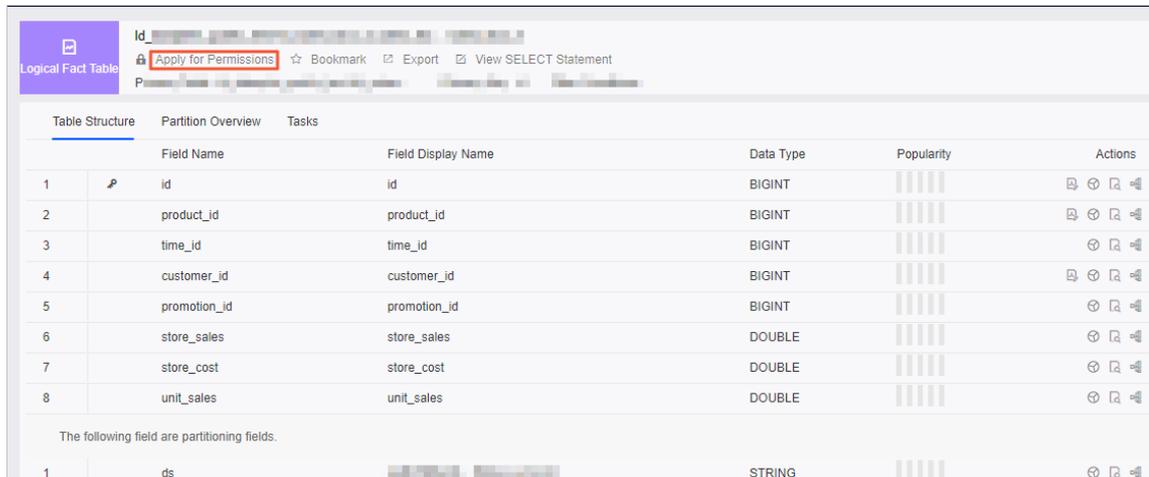
Context

A stream processing task in a project can reference a stream metatable in another project. If a stream processing task that you create needs to reference a stream metatable created by another user or in another project, you must apply for the query, write, update, or delete permission on the stream metatable.

Apply for permissions on a stream metatable

1. [Log on to the Dataphin console](#).
2. Go to the **Stream Metatable Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Stream Metatable Permissions** under **My Permissions** in the left-side navigation pane.
3. On the **Stream Metatable Permissions** tab, go to the **Apply for Stream Metatable Permissions** tab by using one of the following methods:
 - Click **Apply for Stream Metatable Permissions** in the upper-right corner.
 - In the left-side navigation pane, move the pointer over **Stream Metatable Permissions** and click the  icon.

- o In the left-side navigation pane, click the  icon next to My Permissions and select **Stream Metadata Permissions**.
- o In the top navigation bar, click **Map**. On the **Map** page, enter a keyword in the search box to search for tables whose names contain the keyword. Find the stream metadata table on which you want to apply for permissions and click the table name. On the details page that appears, click **Apply for Permissions** in the upper-left part.



4. In the **Select Data Objects** step, set the **Environment**, **Project**, and **Stream Metadata Tables** parameters.
5. Click **Next**.
6. In the **Apply for Permissions** step, set the parameters as required.

Parameter	Description
Authorized Fields	<p>The fields on which you want to apply for permissions. Select fields by using one of the following methods:</p> <ul style="list-style-type: none"> o Enter a keyword in the search box to search for the desired fields and select the fields. o Select fields from the field list of the metadata table.
Permission Type	<p>The permissions that you want to apply for. Valid values:</p> <ul style="list-style-type: none"> o Query o Write o Update o Delete
Account Type	<p>The accounts to be granted the permissions that you are applying for. You can grant the permissions to either or both of the personal account and project production account.</p> <ul style="list-style-type: none"> o The personal account is used for data modeling and development in the development environment and publishing developed objects or tasks to the production environment. o The project production account is used to manage and run tasks that are submitted to the production environment.
Reason for Application	The reason why you apply for the permissions.

7. Click **Submit**.

Query the permissions on a stream metatable

1. On the **Stream Metatable Permissions** tab, find the stream metatable on which you want to query the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, view the fields on which you have applied for permissions and the specific permissions.

Renew the permissions on a stream metatable

1. On the **Stream Metatable Permissions** tab, find the stream metatable on which you want to renew the permissions and click the table name in the **Data Object** column.
2. On the details page that appears, select the fields on which you want to renew permissions.
 - Enter a keyword in the search box to search for the desired fields and select the fields.
 - Select fields from the field list of the metatable.
3. In the lower part of the page, click **Extend Validity**.
4. In the **Apply for Permissions** step, set the parameters as required. For more information, see [Apply for permissions on a stream metatable](#).
5. Click **Submit**.

Remove the permissions on a stream metatable

1. On the **Stream Metatable Permissions** tab, find the stream metatable on which you want to remove the permissions and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.4. Manage permissions on functions

This topic describes how to apply for and remove permissions on functions.

Context

When you use Dataphin for data warehouse modeling, you can use functions to improve the task development efficiency. You can manage the query permission on functions on the **Function Permissions** tab of Dataphin. If a task that you create needs to reference a function created by another user or in another project, you must apply for the query permission on the function.

Apply for the query permission on a function

1. [Log on to the Dataphin console](#).
2. Go to the **Function Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Function Permissions** under **My Permissions** in the left-side navigation pane.
3. On the **Function Permissions** tab, go to the **Apply for Function Permissions** tab by using one of the following methods:
 - Click **Apply for Function Permissions** in the upper-right corner.
 - In the left-side navigation pane, click the  icon next to **Function Permissions**.

- In the left-side navigation pane, click the  icon next to My Permissions and select **Function Permissions**.
4. On the **Apply for Function Permissions** tab, perform the following operations in the configuration wizard:
 - i. In the **Select Data Objects** step, set the **Environment**, **Project**, and **UDF** parameters.
 - ii. Click **Next**.
 - iii. In the **Apply for Permissions** step, set the parameters as required.

Parameter	Description
Permission Type	The permission that you want to apply for. By default, the value Query is selected and you cannot modify the setting.
Account Type	<p>The accounts to be granted the permission that you are applying for. You can grant the permission to either or both of the personal account and project production account.</p> <ul style="list-style-type: none"> ▪ The personal account is used for data modeling and development in the development environment and publishing developed objects or tasks to the production environment. If you select this option, you must also set the Expiration Date parameter. ▪ The project production account is used to manage and run tasks submitted to the production environment. If you select this option, you must also set the Project parameter.
Reason for Application	The reason why you apply for the permission.

5. Click **Submit**.

Remove the query permission on a function

1. On the **Function Permissions** tab, find the function on which you want to remove the query permission and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.5. Manage permissions on data sources

This topic describes how to apply for and remove permissions on data sources.

Context

After you obtain the read and write permissions on a data source, you have the following permissions:

- Permissions to read and write physical tables in the data source but not existing fields in the tables. For more information about how to apply for the read and write permissions on existing fields in the physical tables, see [Manage permissions on physical tables](#).
- Permission to read logical tables in the data source but not existing fields in the tables. For more information about how to apply for the read permission on existing fields in the logical tables, see [Manage permissions on logical tables](#).

Apply for permissions on a data source

1. [Log on to the Dataphin console](#).

2. Go to the **Data Source Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Data Source Permissions** under **My Permissions** in the left-side navigation pane.
3. On the **Data Source Permissions** tab, go to the **Apply for Data Source Permissions** tab by using one of the following methods:
 - o Click **Apply for Data Source Permissions** in the upper-right corner.
 - o In the left-side navigation pane, click the  icon next to **Data Source Permissions**.
 - o In the left-side navigation pane, click the  icon next to **My Permissions** and select **Data Source Permissions**.
4. On the **Apply for Data Source Permissions** tab, set the parameters as required.
 - i. In the **Select Data Objects** step, set the **Environment**, **Project**, and **UDF** parameters.
 - ii. Click **Next**.
 - iii. In the **Apply for Permissions** step, set the parameters as required.

Parameter	Description
Data Source	The data source on which you want to apply for permissions.
Permission Type	The permissions that you want to apply for. Valid values: Read and Write .
Account Type	<p>The accounts to be granted the permissions that you are applying for. You can grant the permissions to either or both of the personal account and project production account.</p> <ul style="list-style-type: none"> ▪ The project production account is used to periodically schedule production tasks in the project. Dataphin uses this type of account to isolate users who have a personal account from the production environment. If you select this option, you must also set the Project parameter. ▪ The personal account is used to perform operations related to data standardization, data modeling, and coding in the development environment. If you select this option, you must also set the Expiration Date parameter.
Reason for Application	The reason why you apply for the permissions.

5. Click **OK**.

Remove the permissions on a data source

1. On the **Data Source Permissions** tab, find the data source on which you want to remove the permissions and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.6. Manage permissions on API operations

You can obtain permissions on API operations only after you activate the **Services** module. This topic describes how to apply for, query, and remove permissions on API operations.

Prerequisites

The Services module is activated.

Apply for permissions on an API operation

1. [Log on to the Dataphin console.](#)
2. Go to the **API Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **API Permissions** under **My Permissions** in the left-side navigation pane.
3. Go to the **API Permission Request** page in Application Center of the Services module by using one of the following methods:
 - On the **API Permissions** tab, click **Apply for API Permissions** in the upper-right corner.
 - In the left-side navigation pane, click the  icon next to **API Permissions**.
 - In the left-side navigation pane, click the  icon next to **My Permissions** and select **API Permissions**.
4. On the **API Permission Request** page, set the parameters in the configuration wizard. For more information, see [Apply for permissions on an API operation](#).

Query permissions on an API operation

1. On the **API Permissions** tab, find the API operation on which you want to query permissions and click the name in the **Data Object** column.
2. On the details page, view the permission information about the API operation.

Remove the permissions on an API operation

1. On the **API Permissions** tab, find the API operation on which you want to remove the permissions and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.1.7. Manage functionality permissions

This topic describes how to apply for and remove functionality permissions.

Apply for a functionality permission

1. [Log on to the Dataphin console.](#)
2. Go to the **Functionality Permissions** tab.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Security** in the top navigation bar.
 - iii. On the **Permissions** page, click **Functionality Permissions** under **My Permissions** in the left-side navigation pane.
3. The **Functionality Permissions** tab does not provide the entry for you to apply for functionality permissions. You can apply for the corresponding functionality permission if you are prompted to apply for the permission when you attempt to open a page, for example, the Overview page of the Data Assets module.
4. Click **Apply for Permission**.

5. In the **Apply for Functionality Permission** dialog box, set the **Expiration Date** and **Reason for Application** parameters.
6. Click **OK**.

Remove a functionality permission

1. On the **Functionality Permissions** tab, find the functionality on which you want to remove the permission and click the  icon in the **Actions** column.
2. In the **Remove Permissions** message, click **OK**.

9.14.3.2. Managed permissions

9.14.3.2.1. Grant and revoke permissions on business units

This topic describes how to grant and revoke permissions on business units.

Grant permissions on a business unit to specific accounts

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Security** in the top navigation bar.
4. On the **Permissions** page, click **Business Unit Permissions** under **Managed Permissions** in the left-side navigation pane.
5. On the **Business Unit Permissions** tab, find the business unit on which you want to grant permissions to specific accounts and click the  icon in the **Actions** column.
6. In the dialog box that appears, set the parameters as required.

Parameter	Description
Grant To	The accounts to which you want to grant permissions on the business unit. You can grant permissions on the business unit to either or both of the personal account and project production account. If you select an account from the Personal Account drop-down list, you must also set the Expiration Date parameter.
Logical Tables	The names of the logical tables on which you want to grant permissions to the specified accounts. You can select logical tables only in the current business unit.
Permission Type	The permissions to be granted. By default, the value Query is selected and you cannot modify the setting.
Reason for Grant of Permissions	The reason why you grant the permissions.

7. Click **OK**.

Revoke permissions on a business unit from a specific account

1. On the **Business Unit Permissions** tab, find the business unit on which you want to revoke permissions from a specific account and click the  icon in the **Actions** column.
2. In the dialog box that appears, set the parameters as required.

Parameter	Description
Revoke From	The account from which you want to revoke permissions on the business unit.
Logical Tables	The names of the logical tables on which you want to revoke permissions from the specified account. You can select logical tables only in the current business unit.
Permission Type	The permissions that you want to revoke. By default, the value Query is selected and you cannot modify the setting.
Reason for Revocation of Permissions	The reason why you revoke the permissions.

3. Click **OK**.

9.14.3.2.2. Grant and revoke permissions on projects

This topic describes how to grant and revoke permissions on projects.

Grant permissions on a project to specific accounts

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Security** in the top navigation bar.
4. On the **Permissions** page, click **Project Permissions** under **Managed Permissions** in the left-side navigation pane.
5. On the **Project Permissions** tab, find the project on which you want to grant permissions to specific accounts and click the  icon in the **Actions** column.
6. In the dialog box that appears, set the parameters as required.

Parameter	Description
Grant To	The accounts to which you want to grant permissions on the project. You can grant permissions on the project to either or both of the personal account and project production account. If you select an account from the Personal Account drop-down list, you must also set the Expiration Date parameter.
Physical Tables	The names of the physical tables on which you want to grant permissions to the specified accounts. You can select physical tables only in the current project.
Permission Type	The permissions to be granted. Valid values: <ul style="list-style-type: none"> ○ Query ○ Write ○ Update ○ Delete
Reason for Grant of Permissions	The reason why you grant the permissions.

7. Click **OK**.

Revoke permissions on a project from a specific account

1. On the **Project Permissions** tab, find the project on which you want to revoke permissions from a specific account and click the  icon in the **Actions** column.
2. In the dialog box that appears, set the parameters as required.

Parameter	Description
Revoke From	The account from which you want to revoke permissions on the project.
Physical Tables	The names of the physical tables on which you want to revoke permissions from the specified account. You can select physical tables only in the current project.
Permission Type	The permissions that you want to revoke.
Reason for Revocation of Permissions	The reason why you revoke the permissions.

3. Click **OK**.

9.14.3.2.3. Grant and revoke permissions on data sources

This topic describes how to grant and revoke permissions on data sources.

Grant permissions on a data source to specific accounts

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Security** in the top navigation bar.
4. On the **Permissions** page, click **Data Source Permissions** under **Managed Permissions** in the left-side navigation pane.
5. On the **Data Source Permissions** tab, find the data source on which you want to grant permissions to specific accounts and click the  icon in the **Actions** column.
6. In the dialog box that appears, set the parameters as required.

Parameter	Description
Grant To	The accounts to which you want to grant permissions on the data source. You can grant permissions on the data source to either or both of the personal account and project production account. If you select an account from the Personal Account drop-down list, you must also set the Expiration Date parameter.
Permission Type	The permissions to be granted. Valid values: <ul style="list-style-type: none">◦ Read◦ Write

Parameter	Description
Reason for Grant of Permissions	The reason why you grant the permissions.

7. Click OK.

Revoke permissions on a data source from a specific account

1. On the **Data Source Permissions** tab, find the data source on which you want to revoke permissions from a specific account and click the  icon in the **Actions** column.
2. In the dialog box that appears, set the parameters as required.

Parameter	Description
Revoke From	The account from which you want to revoke permissions on the data source.
Permission Type	The permissions that you want to revoke.
Reason for Revocation of Permissions	The reason why you revoke the permissions.

3. Click OK.

9.14.3.2.4. Grant and revoke functionality permissions

This topic describes how to grant and revoke functionality permissions.

Grant a functionality permission to a specific account

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Security** in the top navigation bar.
4. On the **Permissions** page, click **Functionality Permissions** under **Managed Permissions** in the left-side navigation pane.
5. On the **Functionality Permissions** tab, find the functionality on which you want to grant the permission to a specific account and click the  icon in the **Actions** column.
6. In the dialog box that appears, set the parameters as required.

Parameter	Description
Grant To	The account to which you want to grant the functionality permission.
Expiration Date	The period in which the functionality permission is valid for the specified account.
Reason for Grant of Permissions	The reason why you grant the functionality permission.

7. Click OK.

Revoke a functionality permission from a specific account

1. On the **Functionality Permissions** tab, find the functionality on which you want to revoke the permission from a specific account and click the  icon in the **Actions** column.
2. In the dialog box that appears, set the parameters as required.

Parameter	Description
Revoke From	The account from which you want to revoke the functionality permission.
Reason for Revocation of Permissions	The reason why you revoke the functionality permission.

3. Click **OK**.

9.14.3.3. Owned permissions

9.14.3.3.1. Configure functionality permissions and transfer the ownership of functionality permissions

This topic describes how to configure functionality permissions and transfer the ownership of functionality permissions.

Configure a functionality permission

1. [Log on to the Dataphin console.](#)
2. Click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Security** in the top navigation bar.
4. On the **Permissions** page, click **Functionality Permission Settings** under **Owned Permissions** in the left-side navigation pane.
5. On the **Functionality Permission Settings** tab, find the functionality for which you want to configure the permission and click the  icon in the **Actions** column.
6. In the **Configure Permission Management** dialog box, specify the user to approve applications for this permission and specify the maximum validity period of the permission.
7. Click **OK**.

Transfer the ownership of a functionality permission

1. On the **Functionality Permission Settings** tab, find the functionality on which you want to transfer the ownership of the permission and click the  icon in the **Actions** column.
2. In the **Transfer Permission Ownership** dialog box, specify the account to which you want to transfer the ownership of the functionality permission.
3. Click **OK**.

9.14.4. Data governance

9.14.4.1. Overview

On the Data Governance page of the Dataphin console, you can analyze resource consumption, control the overall computing and storage costs, and improve service efficiency.

To go to the Data Governance page, perform the following steps:

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Administration** in the top navigation bar. The Data Governance page appears.

The Data Governance page contains the following menus: **Resource Management**, **Governance Overview**, **Governance Console**, **Artifact Management**, and **Recycle Bin**.

- **Resource Management**: includes the **Resource Analysis** page. On the Resource Analysis page, you can view the trends of resource consumption, distribution of resource consumption, and overview of resource governance from a global perspective.
- **Governance Overview**: includes the **Analysis** and **Results** pages. On the **Analysis** page, you can view the diagnosis result of projects for the current tenant and the analysis result of artifacts to be optimized. On the **Results** page, you can view the analysis result of project governance performance. Then, you can evaluate the governance effect and promote the optimization of artifacts.
- **Governance Console**: includes the **My Governance** and **Project Governance** pages. On the **My Governance** page, you can manage and optimize tables or nodes in projects that are owned by the current account. On the **Project Governance** page, you can view the governance information about tables or nodes in projects in which the current account is involved as a member. You can also manage and optimize tables or nodes in projects that are owned by the current account.
- **Artifact Management**: includes the **Metadata Registration**, **Artifact Management**, **Push Management**, and **Task Management** pages. On these pages, you can create artifacts, use custom or system built-in artifacts to configure and initiate push tasks, view the running details and logs of push tasks and artifact tasks, and manage these tasks.
- **Recycle Bin**: allows you to view tables that you have deleted or unpublished in the governance console. Dataphin stores these tables temporarily to prevent misoperation on data.

9.14.4.2. Resource analysis

On the Resource Analysis page, you can view the trends of resource consumption, distribution of resource consumption, and overview of resource governance from a global perspective. This topic describes how to view the global metrics, health status of resources, trend analysis, and project analysis on the Resource Analysis page.

Go to the Resource Analysis page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Resource Analysis** under **Resource Management** in the left-side navigation pane.
5. On the **Resource Analysis** page, you can view the health status of resources, trend analysis, and project analysis.

Global metrics

Global metrics include the **Tables from Services**, **Instances**, **Data Tables**, **Projects**, **Developers**, and **Tables from Computing Engine** parameters.

Parameter	Description
Tables from Services	The number of tables that are synchronized from data sources of specific services to computing engines.
Instances	The total number of task instances with the specified data timestamp.
Data Tables	The total number of tables for the current tenant, including logical tables and physical tables of projects in the production environment and development environment.
Projects	The total number of projects for the current tenant, including projects in the production environment and development environment.
Developers	The total number of Dataphin users who have joined one or more projects of the tenant.
Tables from Computing Engine	The number of tables that are synchronized from computing engines to data sources of specific services.

Resource health status

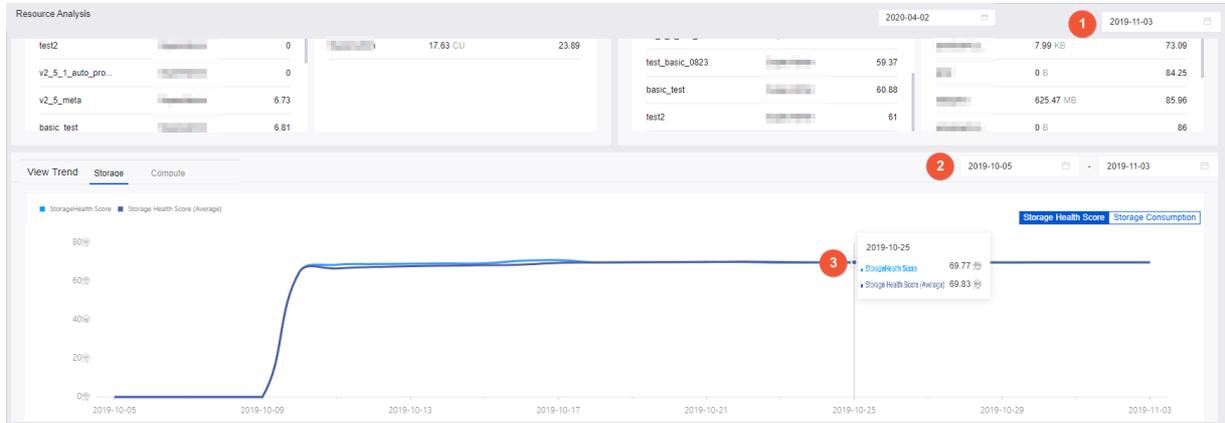
You can view the health status of both computing resources and storage resources.

Parameter	Description
Compute Health Score	<p>The score for the health status of computing resources with the specified data timestamp. The health scores are graded into the following three levels:</p> <ul style="list-style-type: none"> • High: The health score is greater than or equal to 85. The message Your score indicates that you excel in data asset management. Keep up the good work is displayed. • Medium: The health score is greater than or equal to 60 and less than 85. The message Your score is slightly higher than the threshold. We recommend that you continue to optimize data asset management is displayed. • Low: The health score is less than 60. The message Your score is lower than the threshold. We recommend that you optimize data asset management at your earliest opportunity is displayed.

Parameter	Description
Consumed Compute Resources	<p>The amount of consumed computing resources. Unit: CU, KCU, or CM .</p> <ul style="list-style-type: none"> 1 core × 60 seconds = 1 CM 24 hours × 60 minutes × 1 CM = 1 CU 1,000 CU = 1 KCU <p>Different units are used in the following situations:</p> <ul style="list-style-type: none"> If the amount of consumed computing resources is less than 1,000 CM, the unit CM is used, for example, 987 CM. If the amount of consumed computing resources is greater than 1,000 CM, the unit CU is used, for example, 0.83 CU , which equals to 1,200 CM. If the amount of consumed computing resources is greater than 1,000 CU, the unit KCU is used, for example, 1.1 KCU , which equals to 1,100 CU. <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> The value is accurate to two decimal places. You can convert the consumption value to a new one with a larger unit as needed. The amount of consumed computing resources is calculated based on the total execution time for which the CPU was dedicated to underlying tasks. For example, if a task occupies one CPU core and runs for one day, one CU is consumed. </div>
Storage Health Score	The score for the health status of storage resources with the specified data timestamp.
Storage Consumption	<p>The amount of consumed storage resources. Unit: byte, KB, MB, GB, TB, PB, EB, or ZB.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p>Note</p> <ul style="list-style-type: none"> The following formulas describe the conversion from the unit of byte to the units of KB, MB, GB, TB, PB, EB, and ZB for the measurement of consumed storage resources: 1024 B => 1 KB 1024 * 1024 B => 1 MB 1024 * 1024 * 1024 B => 1 GB 1024 * 1024 * 1024 * 1024 B => 1 TB 1024 * 1024 * 1024 * 1024 * 1024 B => 1 PB 1024 * 1024 * 1024 * 1024 * 1024 * 1024 B => 1 EB 1024 * 1024 * 1024 * 1024 * 1024 * 1024 * 1024 B => 1 ZB </div>
Projects (Instant Optimization)	The top 10 projects with the lowest health scores in computing or those with the lowest health scores in storage.
Owners (Instant Optimization)	The top 10 owners with the lowest health scores in computing or those with the lowest health scores in storage.

Trend analysis

In the View Trend section, you can view information about global computing resources and storage resources in a specific time period, including the trends of the health score in storage, storage resource consumption, health score in computing, and computing resource consumption. The trend charts help you analyze data stability.

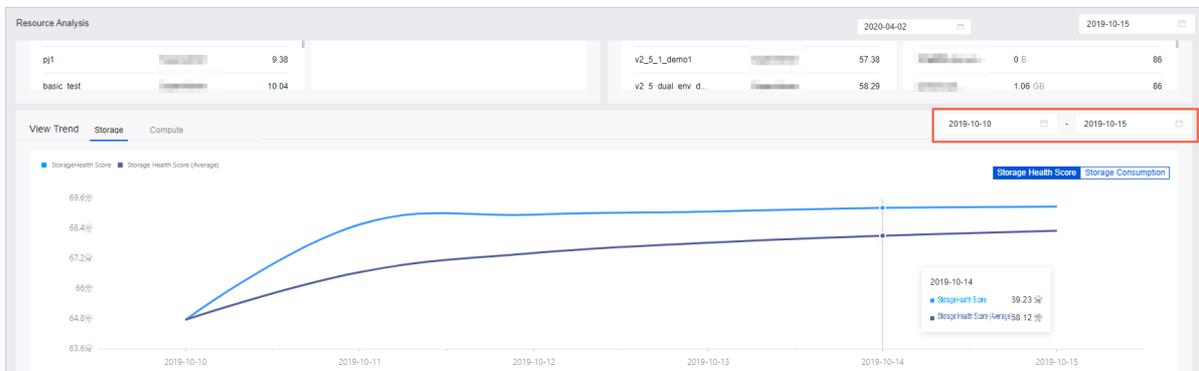


The base date, marked as 1 in the preceding figure, is displayed in the upper-right corner of the page. By default, the trend chart shows the trends of the corresponding data in the 30 days prior to the base date.

- Click the **Storage** tab in the **View Trend** section to view the trend chart of the health score in storage. By default, the trend chart shows the trends of the health score in storage and the average score in the 30 days prior to the base date.

Move the pointer over a point of time in the chart. You can view the health score in storage corresponding to the specified date and the average score in the seven days prior to the specified date. The data stability is determined based on the difference between the health score in storage and the average score. A smaller difference indicates a higher data stability. For example, at the time point marked as 3 in the preceding figure, the health score in storage of October 25, 2019 is 69.77 and the average score in the seven days prior to October 25, 2019 is 69.83.

You can specify a time period, marked as 2 in the preceding figure, based on business requirements to view the corresponding trends of the health score in storage and the average score. For example, you can specify a time period from October 10, 2019 to October 15, 2019. The trend chart shows the trends of the health score in storage and the average score in this time period, as shown in the following figure. If you move the pointer over the time point of October 14, 2019 in the chart, the corresponding scores appear, that is, 69.23 for the health score in storage on October 14, 2019 and 68.12 for the average score in the seven days prior to October 14, 2019.



- On the **Storage** tab in the **View Trend** section, click **Storage Consumption** in the upper-right corner. By default, the trend chart shows the trends of storage resource consumption and the average consumption in the 30 days prior to the base date.

 **Note** The data on the Storage Consumption chart is presented in the same way as that on the Storage Health Score chart.

- Click the **Compute** tab in the **View Trend** section to view the trend chart of the health score in computing. By default, the trend chart shows the trends of the health score in computing and the average score in the 30 days prior to the base date.

 **Note** The data on the Compute Health Score chart is presented in the same way as that on the Storage Health Score chart.

- On the Compute tab in the View Trend section, click **Consumed Compute Resources** in the upper-right corner. By default, the trend chart shows the trends of computing resource consumption and the average consumption in the 30 days prior to the base date.

 **Note** The data on the Consumed Compute Resources chart is presented in the same way as that on the Storage Health Score chart.

Project analysis

In the **Project Analysis** section, you can view the analysis result of projects. You can also filter projects to find a desired project and view the analysis result of the project. View the analysis result of projects in the following ways:

- View the analysis results of the projects that are displayed by default, as described in the following table.

Parameter	Description
Project Name	The name of the project.
Business Unit	The business unit to which the project belongs.
Instances	The number of task instances in the project, including instances of sync tasks and code tasks.
Data Tables	The number of tables in the project, including physical tables and logical tables.
Consumed Compute Resources	The amount of computing resources that are consumed by all tasks in the project. Unit: CU.
Consumed Storage Resources	The amount of storage resources that are consumed by all tables in the project.
Compute Health Score	The health score in computing of the project.
Storage Health Score	The health score in storage of the project.
Administrator	The administrator of the project.
Artifacts	The number of governance items to be optimized.

- Enter a keyword in the search box to search for projects whose names contain the keyword. You can also click the  icon and set the **Environment** and **Business Unit** parameters to find desired projects and view the analysis result of these projects.

9.14.4.3. Governance overview

9.14.4.3.1. Governance analysis

On the Analysis page, you can view the diagnosis result of projects for the current tenant and the analysis result of artifacts to be optimized. This topic describes the information you can view on the Analysis page.

Go to the Analysis page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Analysis** under **Governance Overview** in the left-side navigation pane.
5. On the **Analysis** page, click the **Storage** and **Compute** tabs to view the health scores in storage and computing, score details, and project governance details.

In the upper-right corner of the Analysis page, specify a base date to display the corresponding business data.

Resource categories

By default, the Analysis page displays the health score in storage and score details of projects for the current tenant. You can click the **Compute** tab on the **Analysis** page to view the health score in computing and score details. You can view the following information on the Storage or Compute tab:

- Click the **Storage** tab on the **Analysis** page. The health score in storage appears on the left of the page. The **Score Details** section on the right of the page displays the numbers of artifacts, optimized governance items, and tables to be optimized. The Score Details section also lists information about each artifact, as described in the following table.

Parameter	Description
Artifact Name	The name of the artifact. Artifacts for storage include Excessively Long Retention Period, No Management Policies Specified, Idle Physical Tables, Empty Table, No Management Policies Specified for Logical Tables, Idle Logical Tables, Empty Logical Table, and Logical Table Conversion (Empty Physical Tables).
Score Point Reduction	The points deducted for the occurrence of the artifact.
Storage Resources (To Be Optimized)	The amount of storage resources to be optimized.
Tables (To Be Optimized)	The number of tables to be optimized.

- Click the **Compute** tab on the **Analysis** page. The health score in computing appears on the left of the page. The **Score Details** section on the right of the page displays the numbers of artifacts, optimized governance items, and nodes to be optimized. The Score Details section also lists information about each artifact, as described in the following table.

Parameter	Description
-----------	-------------

Parameter	Description
Artifact Name	The name of the artifact. Artifacts for computing include Unread Output Table, Inappropriate Data Scanning, Unexpected Large Amounts of Data, Data Skew, Failed Task Node, No Imported Data, and No Input Data.
Score Point Reduction	The points deducted for the occurrence of the artifact based on the built-in health assessment model.
Compute Resources (To Be Optimized)	The amount of computing resources to be optimized.
Nodes (To Be Optimized)	The number of nodes to be optimized.

Project governance details

In the Project Governance Details section of the Storage and Compute tabs, you can view the governance information about each project in the corresponding business unit, such as the amount of consumed resources, the ratio of optimized governance items to total governance items, and the number of governance items to be optimized. You can click the **Compute** or **Storage** tab on the **Analysis** page to view the governance information about the computing resources or storage resources of projects. You can also filter projects to find a desired project and view the governance details of the project. View the governance details of projects in the following ways:

- Click the **Storage** tab to view the governance information about the storage resources of projects, as described in the following table.

Parameter	Description
Project Name	The name of the project.
Business Unit	The business unit to which the project belongs.
Administrator	The administrator of the project.
Consumed Storage Resources	The amount of storage resources that are consumed by all tables in the project.
Rate of Optimized Storage Resources	The ratio of optimized governance items to total governance items that were identified in the seven days prior to the date you specify in the upper-right corner of the page.
Artifacts	The number of governance items to be optimized in the project.

Parameter	Description
Actions	<p>The entries for you to view data trends and health score rankings.</p> <ul style="list-style-type: none"> Click the  icon in the Actions column of a project. In the View Trend dialog box, you can view the trend chart of the health score in storage and storage resource consumption in the last 7 or 30 days. You can also move the pointer over a point of time in the chart to view the health score in storage and storage resource consumption of the specified date. Click the  icon in the Actions column of a project. In the View Rankings dialog box, you can view the ratio of the storage resources that each table consumes in the project. You can also view the rankings of tables based on the storage resource consumption in descending order or based on the health score in storage in ascending order.

- Click the **Compute** tab to view the governance information about the computing resources of projects, as described in the following table.

Parameter	Description
Project Name	The name of the project.
Business Unit	The business unit to which the project belongs.
Administrator	The administrator of the project.
Consumed Compute Resources	The amount of computing resources that are consumed by all nodes in the project.
Rate of Optimized Compute Resources	The ratio of optimized governance items to total governance items that were identified in the seven days prior to the date you specify in the upper-right corner of the page.
Artifacts	The number of governance items to be optimized in the project.
Actions	<p>The entries for you to view data trends and health score rankings.</p> <ul style="list-style-type: none"> Click the  icon in the Actions column of a project. In the View Trend dialog box, you can view the trend chart of the health score in computing and computing resource consumption in the last 7 or 30 days. You can also move the pointer over a point of time in the chart to view the health score in computing and computing resource consumption of the specified date. Click the  icon in the Actions column of a project. In the View Rankings dialog box, you can view the ratio of the computing resources that each node consumes in the project. You can also view the rankings of nodes based on the computing resource consumption in descending order or based on the health score in computing in ascending order.

- Enter a keyword in the search box to search for projects whose names contain the keyword. You can also

click the  icon and set **Environment**, **Business Unit**, and **Owner** to find desired projects and view the governance details of these projects.

9.14.4.3.2. Governance results

On the Results page, you can view the analysis result of project governance performance. Then, you can evaluate the governance effect and promote the optimization of artifacts. This topic describes how to view the overall governance metrics, trends, and project governance details on the Results page.

Go to the Results page

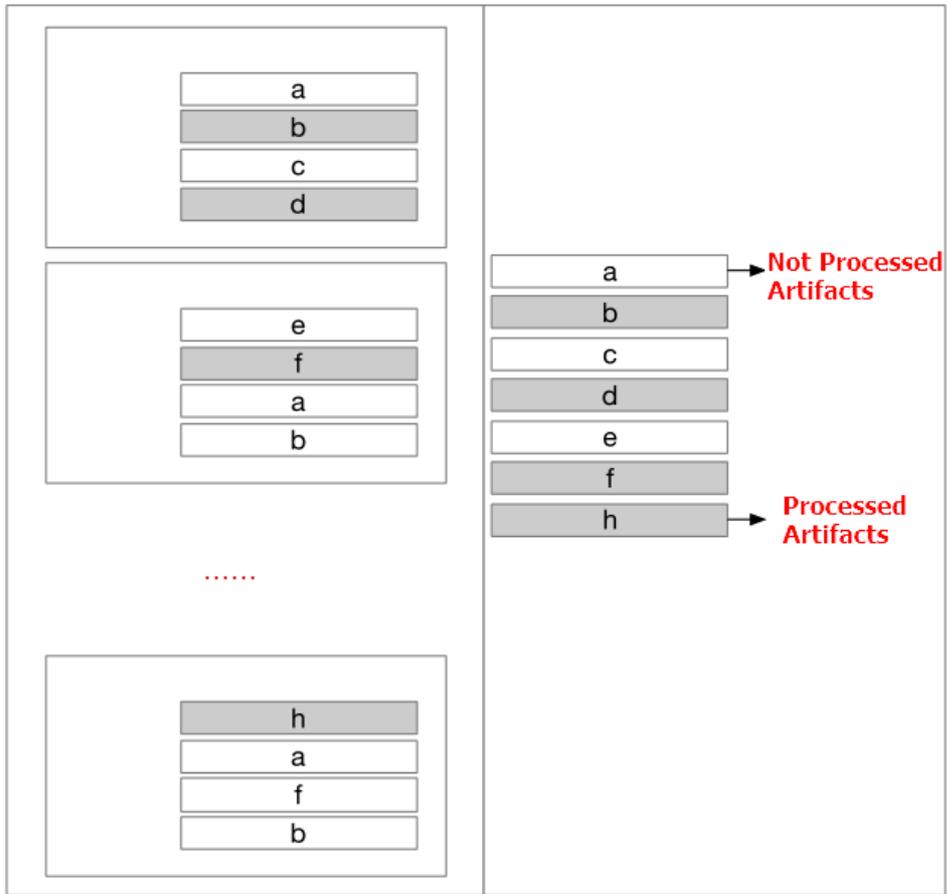
1. [Log on to the Dataphin console.](#)
2. Go to the **Data Governance** page.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Administration** in the top navigation bar.
3. On the **Data Governance** page, click **Results** under **Governance Overview** in the left-side navigation pane.

In the upper-right corner of the **Results** page, specify a base date to display the corresponding business data. The Results page consists of the following sections: **Artifact Details**, **Saved Resources**, **View Trend**, and **Project Governance Details**.

Overall governance metrics

The overall governance metrics include two categories: **Artifact Details** and **Saved Resources**. The **Artifact Details** section displays the **Rate of Processed Resources** and **Processed Artifacts** metrics. The **Saved Resources** section displays the **Compute** and **Storage** metrics.

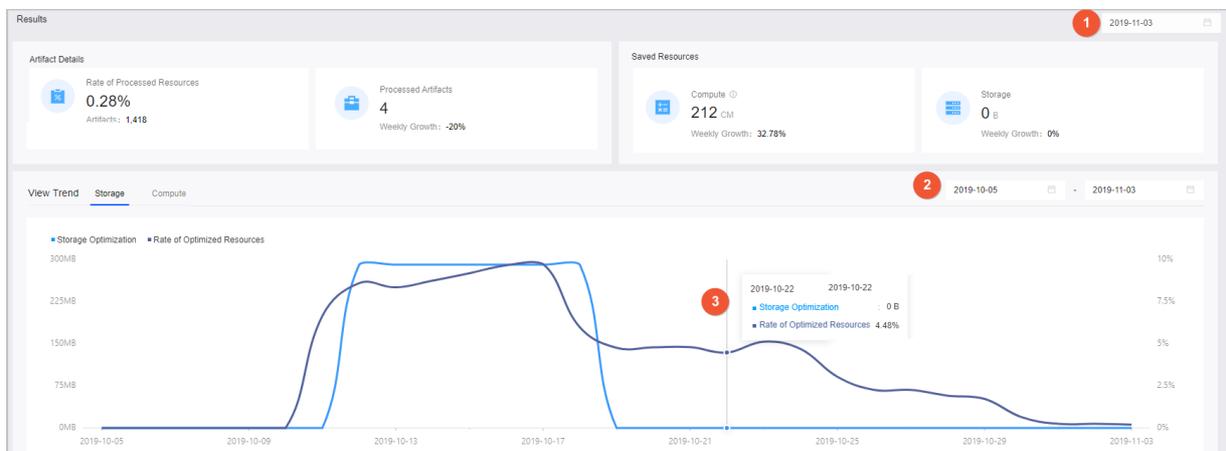
- **Rate of Processed Resources**: the ratio of optimized governance items to total governance items that were identified in the last seven days.



- **Processed Artifacts:** the number of governance items that were optimized in the last seven days.
- **Compute:** the amount of computing resources that have been saved by optimized governance items of the computing type.
- **Storage:** the amount of storage resources that have been saved by optimized governance items of the storage type.

Trend analysis

In the View Trend section, you can view information about global computing resources and storage resources in a specific time period, including the trends of the amount of saved resources and the ratio of optimized governance items to total governance items.

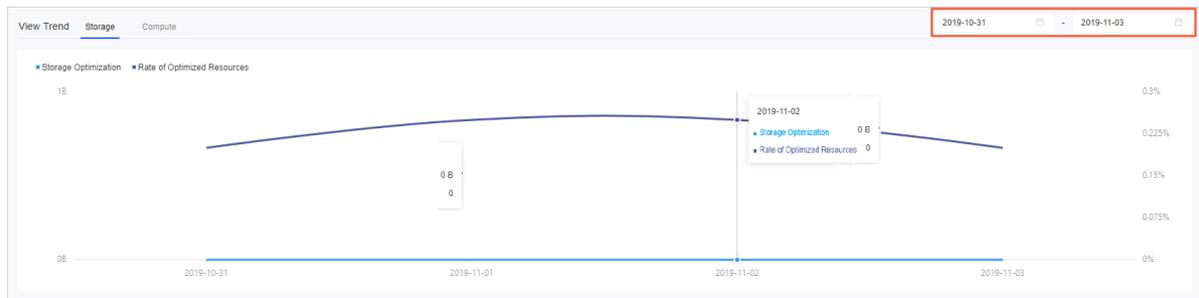


The base date, marked as 1 in the preceding figure, is displayed in the upper-right corner of the page. By default, the trend chart shows the trends of the corresponding data in the 30 days prior to the base date.

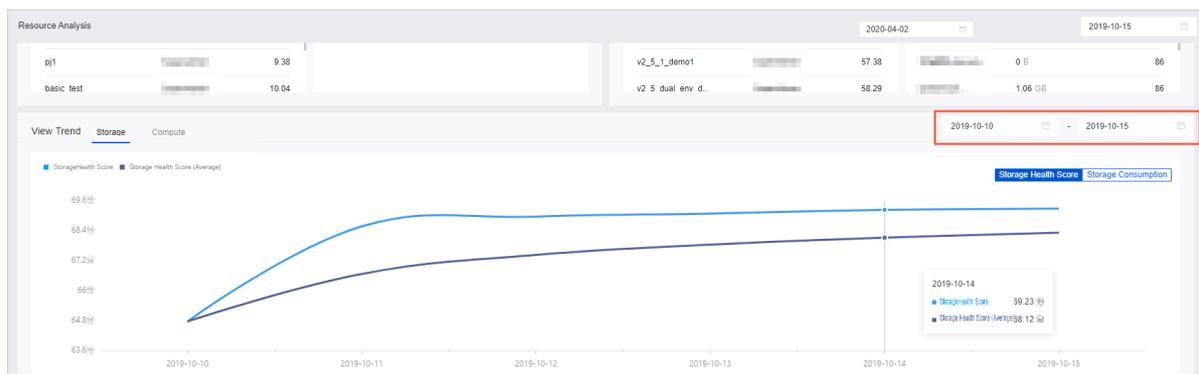
- Click the **Storage** tab in the **View Trend** section to view the trends of the amount of saved storage resources and the ratio of optimized governance items of the storage type to total governance items of the storage type in the 30 days prior to the base date.

Move the pointer over a point of time in the chart. You can view the ratio of optimized governance items of the storage type to total governance items of the storage type corresponding to the specified date and the amount of storage resources that have been saved in the seven days prior to the specified date. For example, at the time point marked as 3 in the preceding figure, the amount of storage resources that have been saved in the seven days prior to October 22, 2019 is 0 bytes and the ratio of optimized governance items of the storage type to total governance items of the storage type on that day is 4.48%.

You can specify a time period, marked as 2 in the preceding figure, based on business requirements to view the corresponding trends of the ratio of optimized governance items of the storage type to total governance items of the storage type and the amount of saved storage resources. For example, you can specify a time period from October 31, 2019 to November 3, 2019. The trend chart shows the trends of the ratio of optimized governance items of the storage type to total governance items of the storage type and the amount of saved storage resources in this time period, as shown in the following figure. If you move the pointer over the time point of November 2, 2019 in the chart, the corresponding data appears, that is, 0.0% for the ratio of optimized governance items of the storage type to total governance items of the storage type on November 2, 2019 and 0 bytes for the amount of storage resources that have been saved in the seven days prior to November 2, 2019.



- Click the **Compute** tab in the **View Trend** section to view the trends of the amount of saved computing resources and the ratio of optimized governance items of the computing type to the total governance items of the computing type in the 30 days prior to the base date.



Note The data on the Compute tab is presented in the same way as that on the Storage tab.

Project governance details

In the **Project Governance Details** section, you can view the governance information about each project in the corresponding business unit, such as the amount of consumed storage or computing resources, the health score in storage or computing, and the amount of saved storage or computing resources.

Project Governance Details									
Project Name	Business Unit	Owner	Consumed Compute Resources	Compute Health Score	Compute Optimization	Consumed Storage Resources	Storage Health Score	Storage Optimization	Actions
dp_gov			141.67 CM	49.87	212 CM	1.05 TB	77.9	0 B	
dp_gov_dev			0 CM	0	0 CM	17.34 GB	66.87	0 B	
dataphin_online_test			23 CM	9.33	0 CM	422.3 MB	68.83	0 B	
demo			109 CM	3.68	0 CM	20.86 MB	81.36	0 B	

Parameter	Description
Project Name	The name of the project.
Business Unit	The business unit to which the project belongs.
Owner	The owner of the project.
Consumed Storage Resources	The amount of storage resources that are consumed by all tables in the project.
Storage Health Score	The health score in storage of the project.
Consumed Compute Resources	The amount of computing resources that are consumed to run MaxCompute tasks in the project, including SQL tasks that are created by developers and automatically generated by Dataphin.
Compute Health Score	The health score in computing of the project.
Storage Optimization	The amount of storage resources that have been saved in the project by optimized governance items of the storage type.
Compute Optimization	The amount of computing resources that have been saved in the project by optimized governance items of the computing type.

Parameter	Description																																													
<p>Actions</p>	<p>The entries for you to view data trends and health score rankings.</p> <ul style="list-style-type: none"> • Click the  icon in the Actions column of a project. In the View Trend dialog box, you can view the trend chart of the corresponding data in the last 7 or 30 days. <ul style="list-style-type: none"> ◦ On the Storage tab, move the pointer over a point of time in the chart. You can view the health score in storage and storage resource consumption of the specified date. ◦ Click the Compute tab. On the Compute tab, move the pointer over a point of time in the chart. You can view the health score in computing and computing resource consumption of the specified date. • Click the  icon in the Actions column of a project. In the View Rankings dialog box, you can view the rankings of project owners based on their project governance performance. <ul style="list-style-type: none"> ◦ On the Storage tab, you can view the rankings of project owners based on the health score in storage, amount of consumed storage resources, and amount of saved storage resources. <div data-bbox="563 864 1345 1319" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> View Rankings Storage Compute ✕ </div> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 10%;">View Rankings</th> <th style="width: 20%;">Owner</th> <th style="width: 20%;">Storage Health Score</th> <th style="width: 10%;">Storage Optimization</th> <th style="width: 30%;">Consumed Storage Resources</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>77.19</td> <td>0</td> <td>486.42 GB</td> </tr> <tr> <td>2</td> <td></td> <td>84.94</td> <td>0</td> <td>4.17 GB</td> </tr> <tr> <td>3</td> <td></td> <td>62.13</td> <td>0</td> <td>1.56 GB</td> </tr> <tr> <td>4</td> <td></td> <td>86</td> <td>0</td> <td>580.86 GB</td> </tr> </tbody> </table> <div style="text-align: right; padding-top: 10px;"> Confirm </div> </div> ◦ Click the Compute tab. On the Compute tab, you can view the rankings of project owners based on the health score in computing, amount of consumed computing resources, and amount of saved computing resources. <div data-bbox="563 1440 1345 1816" style="border: 1px solid #ccc; padding: 5px; margin: 10px 0;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid #ccc; padding-bottom: 5px;"> View Rankings Storage Compute ✕ </div> <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 10%;">View Rankings</th> <th style="width: 20%;">Owner</th> <th style="width: 20%;">Compute Health Score</th> <th style="width: 10%;">Compute Optimization</th> <th style="width: 30%;">Consumed Compute Resources</th> </tr> </thead> <tbody> <tr> <td>1</td> <td></td> <td>6.59</td> <td>0</td> <td>126.67 CM</td> </tr> <tr> <td>2</td> <td></td> <td>50.68</td> <td>0</td> <td>6.67 CM</td> </tr> <tr> <td>3</td> <td></td> <td>100</td> <td>0</td> <td>8.33 CM</td> </tr> </tbody> </table> <div style="text-align: right; padding-top: 10px;"> Confirm </div> </div>	View Rankings	Owner	Storage Health Score	Storage Optimization	Consumed Storage Resources	1		77.19	0	486.42 GB	2		84.94	0	4.17 GB	3		62.13	0	1.56 GB	4		86	0	580.86 GB	View Rankings	Owner	Compute Health Score	Compute Optimization	Consumed Compute Resources	1		6.59	0	126.67 CM	2		50.68	0	6.67 CM	3		100	0	8.33 CM
View Rankings	Owner	Storage Health Score	Storage Optimization	Consumed Storage Resources																																										
1		77.19	0	486.42 GB																																										
2		84.94	0	4.17 GB																																										
3		62.13	0	1.56 GB																																										
4		86	0	580.86 GB																																										
View Rankings	Owner	Compute Health Score	Compute Optimization	Consumed Compute Resources																																										
1		6.59	0	126.67 CM																																										
2		50.68	0	6.67 CM																																										
3		100	0	8.33 CM																																										

Enter a keyword in the search box to search for projects whose names contain the keyword. You can also click the  icon and set **Environment**, **Business Unit**, and **Owner** to find desired projects and view the governance details of these projects.

9.14.4.4. Governance console

In the governance console, you can view the governance information about tables or nodes in projects in which you are involved as a member. You can also manage and optimize tables or nodes in projects in which you are involved as an administrator.

As the administrator or developer of a project, you can filter tables or nodes that require optimization by selecting the governance scope, governance target, and artifact, and then view corresponding information and manage and optimize governance items in the governance console. The features provided on the **My Governance** page are similar to those on the **Project Governance** page. The following example demonstrates how to use the governance console. The **My Governance** page is used in this example.

Go to the My Governance page

1. [Log on to the Dataphin console](#).
2. Go to the **My Governance** page.
 - i. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
 - ii. On the **Data Assets** page, click **Administration** in the top navigation bar.
 - iii. On the **Data Governance** page, click **My Governance** under **Governance Console** in the left-side navigation pane.

- iv. On the **My Governance** page, set the **Governance Scope**, **Governance Target**, and **Artifact** parameters to filter governance items.

Parameter	Description
Governance Scope	The business scope of the governance target. Valid values: Storage and Compute .
Governance Target	<ul style="list-style-type: none"> ▪ If you select Storage for the Governance Scope parameter, the valid values of the Governance Target parameter are Physical Table and Logical Table. ▪ If you select Compute for the Governance Scope parameter, the value of the Governance Target parameter is fixed to Task Node.
Artifact	<ul style="list-style-type: none"> ▪ If you select Storage for the Governance Scope parameter and select Physical Table for the Governance Target parameter, the valid values of the Artifact parameter are No Management Policies Specified, Empty Table, Idle Physical Tables, Excessively Long Retention Period, and custom artifacts. ▪ If you select Storage for the Governance Scope parameter and select Logical Table for the Governance Target parameter, the valid values of the Artifact parameter are No Management Policies Specified for Logical Tables, Idle Logical Tables, Empty Logical Table, Logical Table Conversion (Empty Physical Tables), and custom artifacts. ▪ If you select Compute for the Governance Scope parameter and select Task Node for the Governance Target parameter, the valid values of the Artifact parameter are No Input Data, No Imported Data, Failed Task Node, Data Skew, Unexpected Large Amounts of Data, Inappropriate Data Scanning, and Unread Output Table.

- v. View the details of governance items.

- If you select **Storage** for the **Governance Scope** parameter and select **Physical Table** for the **Governance Target** parameter, you can view information about each physical table to be optimized.

Parameter	Description
Table Name	The name of the physical table.
Project Name	The name of the project to which the physical table belongs.
Business Unit	The business unit to which the physical table belongs.
Owner	The owner of the physical table.
Health Score	The health score of the physical table.
Storage Space	The size of MaxCompute storage resources that the physical table occupies.
Retention Period	The retention period of the physical table.
Number of Days (Accessed from First Day to Last Day in the Last 33 Days)	The number of visits to the physical table in the last 33 days.
Processing Status	The processing status of the node. Valid values: Processed and Not Processed .

- If you select **Storage** for the **Governance Scope** parameter and select **Logical Table** for the **Governance Target** parameter, you can view information about each logical table to be optimized.

Parameter	Description
Model Name	The name of the logical table.
Project Name	The name of the project to which the logical table belongs.
Business Unit	The business unit to which the logical table belongs.
Owner	The owner of the logical table.
Health Score	The health score of the logical table.
Storage Space	The size of MaxCompute storage resources that the logical table occupies.
Retention Period	The retention period of the logical table.
Environment	The environment to which the logical table belongs. Valid values: Development and Production .
Processing Status	The processing status of the node. Valid values: Processed and Not Processed .

- If you select **Compute** for the **Governance Scope** parameter and select **Task Node** for the **Governance Target** parameter, you can view information about each node to be optimized.

Parameter	Description
Node Name	The name of the node.
Node ID	The ID of the node.
Node Type	The type of the node.
Project Name	The name of the project to which the node belongs.
Owner	The owner of the node.
Health Score	The health score of the node.
CPU Consumption	The amount of computing resources that the node consumes.
Duration	The duration during which the node instance is run.
Processing Status	The processing status of the node. Valid values: Processed and Not Processed .

Manage physical tables

The supported actions on physical tables include **Move to Recycle Bin**, **Set Retention Period**, and **Pause Node**.

1. On the **My Governance** page, you can find physical tables to be optimized by using one of the following methods:
 - Select **Storage** for the **Governance Scope** parameter and select **Physical Table** for the **Governance Target** parameter to filter physical tables.
 - Click the  icon next to **Select a project** and select a project to filter physical tables.
 - Enter a keyword in the search box to search for physical tables whose names contain the keyword.
 - Click the  icon next to **All** and select **All**, **Processed**, or **Not Processed** to filter physical tables.
2. Manage physical tables by using one of the following methods:
 - Find the physical table that you want to manage and click the  icon in the **Actions** column to move the physical table to the recycle bin.
You can also select multiple physical tables and click **Move to Recycle Bin** in the lower part of the page to move them to the recycle bin. In the **Confirm** message, click **Confirm**. The selected physical tables are moved to the recycle bin.
 - Find the physical table that you want to manage and click the  icon in the **Actions** column. In the **Set Retention Period** dialog box, set a retention period and click **OK**.
You can also select multiple physical tables and click **Set Retention Period** in the lower part of the page. In the **Set Retention Period** dialog box, set a retention period and click **OK**. A retention period is set for the selected physical tables.
 - Find the physical table that you want to manage and click the  icon in the **Actions** column. In the **Confirm** message, click **Confirm**. The system stops using this physical table.

 **Note** The system does not allow you to resume a paused physical table.

Manage logical tables

Only the **Set Retention Period** action is supported on logical tables.

1. On the **My Governance** page, you can find logical tables to be optimized by using one of the following methods:
 - Select **Storage** for the **Governance Scope** parameter and select **Logical Table** for the **Governance Target** parameter to filter logical tables.
 - Click the  icon next to **Select a project** and select a project to filter logical tables.
 - Enter a keyword in the search box to search for logical tables whose names contain the keyword.
 - Click the  icon next to **All** and select **All**, **Processed**, or **Not Processed** to filter logical tables.
2. Find the logical table that you want to manage and click the  icon in the **Actions** column. In the **Set Retention Period** dialog box, set a retention period and click **OK**.

You can also select multiple logical tables and click **Set Retention Period** in the lower part of the page. In the **Set Retention Period** dialog box, set a retention period and click **OK**. A retention period is set for the selected logical tables.

Manage nodes

The supported actions on nodes include **Pause Node** and **Unpublish Node**.

1. On the **My Governance** page, you can find nodes to be optimized by using one of the following methods:
 - Select **Compute** for the **Governance Scope** parameter and select **Task Node** for the **Governance Target** parameter to filter nodes.
 - Click the  icon next to **Select a project** to filter nodes.
 - Enter a keyword in the search box to search for nodes whose names contain the keyword.
 - Click the  icon next to **All** and select **All**, **Processed**, or **Not Processed** to filter nodes.
2. Manage nodes by using one of the following methods:
 - Find the node that you want to manage and click the  icon in the **Actions** column. In the **Confirm** message, click **Confirm**. The system stops running this node.

 **Note** The system does not allow you to resume a paused node.

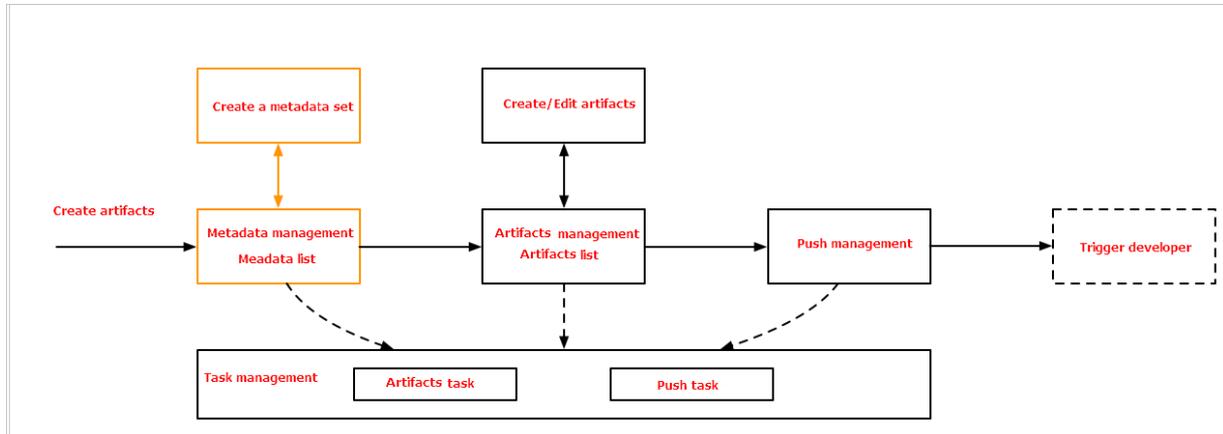
- Find the node that you want to manage and click the  icon in the **Actions** column. In the **Confirm** message, click **Confirm**. The node is not run in the production environment as scheduled.

9.14.4.5. Artifact management

9.14.4.5.1. Artifact management process

The Artifact Management module allows you to register metadata sets, create artifacts and push tasks, view the running details and logs of push tasks and artifact tasks, and manage these tasks. In the Artifact Management module, you can also manage custom artifacts and system built-in artifacts.

Dataphin allows you to create custom artifacts based on your business requirements and use the governance settings in data asset management.



The artifact management process involves the following features:

1. Metadata registration: allows you to create and manage metadata sets that are required for data governance.
2. Artifact management: allows you to create and manage artifacts based on the configured metadata sets. Artifacts define the identification rules and policies for data governance.
3. Push management: allows you to create push tasks so that you can receive push notifications when occurrences of one or more specified artifacts are identified for data objects, including tables and nodes, in the selected projects. The push notifications instruct you to process the data objects that require optimization.
4. Task management: allows you to manage artifact tasks and push tasks.

9.14.4.5.2. Metadata registration

On the Metadata Registration page, you can create and manage metadata sets that are required for data governance. This topic describes how to create, edit, check, and delete metadata sets, and change owners of metadata sets.

Go to the Metadata Registration page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Metadata Registration** under **Artifact Management** in the left-side navigation pane.
5. On the **Metadata Registration** page, view information about registered metadata sets.

Parameter	Description
Metadata Set Name	The name of the metadata set.
Data Source Type	The type of the data source for the metadata set.

Parameter	Description
Data Object	The type of the data object that is managed based on the metadata set. Valid values: Physical Table , Logical Table , and Task Node .
Owner	The owner of the metadata set.
Metrics	The number of metrics that are registered in the metadata set.
Referenced by Artifacts	Indicates whether the metadata set is referenced and the number of times that the metadata set is referenced. For example, the value is Yes if you reference a metric in the metadata set when you create an artifact.
Check Status	The check result of the metadata set. Valid values: Succeeded and Failed . If any fields that the metadata set references are missing or the referenced table is renamed, the check fails. If all fields that the metadata set references exist and the metadata set can properly connect to the referenced table, the check is successful.

You can enter a keyword in the search box to search for metadata sets whose names contain the keyword. You can also click the  icon and set the **Data Object**, **Data Source Type**, **Referenced Status**, and **Owner** parameters to filter metadata sets.

Create a metadata set

1. On the **Metadata Registration** page, click **Create Metadata Set** in the upper-right corner.
2. In the **Create Metadata Set** dialog box, set the parameters as required.

Create Metadata Set
✕

Data Source

* Metadata Set Name:

* Data Source Type: * Project:

Metrics

* Data Table Name: * Data Object:

Metric Registration + Add Field

Field Name	Metric Name	Data Type	Metric Type	Connectivity/ Actions
Availability				

Section	Parameter	Description
	Metadata Set Name	The name of the metadata set. Limits: <ul style="list-style-type: none"> ◦ The name must be 1 to 30 characters in length. ◦ The name must be unique.
	Data Source Type	The type of the data source for the metadata set. Select MAX_COMPUTE .

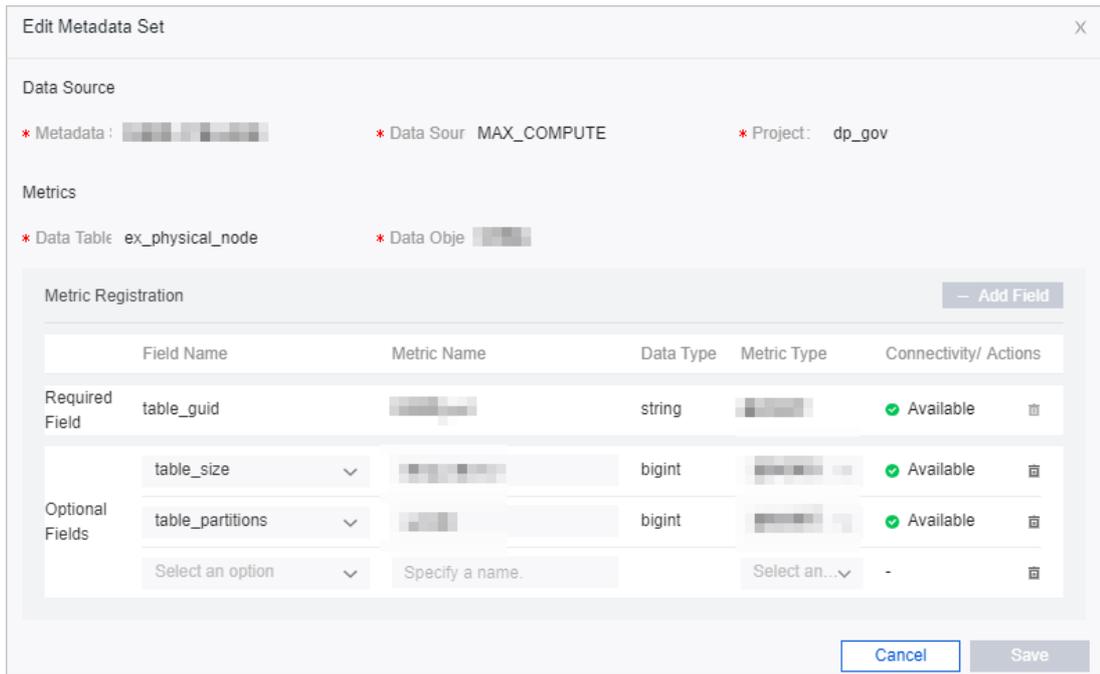
Data Source Section	Parameter	Description
	Project	<p>The data source of the metadata set. Select a computing engine from the drop-down list or search for it by entering a keyword in the search box.</p> <p>Note The drop-down list displays the computing engines that are created on the Planning page of the Dataphin console.</p>
Metrics	Data Table Name	<p>The table that is referenced by the metadata set. Select a table from the drop-down list or search for it by entering a keyword in the search box.</p> <p>Note</p> <ul style="list-style-type: none"> The drop-down list displays the physical tables, logical tables, and nodes that belong to the project of the selected computing engine. You can create only one metadata set based on each table.
	Data Object	<p>The type of the data object to be managed based on the metadata set. Valid values: Physical Table, Logical Table, and Task Node.</p>
	Metric Registration	<p>The metrics that are registered in the metadata set. At least one field of each table must be registered in the metadata set. You can click Add Field to add more fields. For more information, see Edit a metadata set.</p>

3. Click **Save**. The metadata set is created.

Edit a metadata set

- On the **Metadata Registration** page, find the metadata set that you want to edit and click the  icon in the **Actions** column.
- In the **Edit Metadata Set** dialog box, add or remove fields as required. The other parameters cannot be modified.
 - Add fields to the metadata set.

a. Click **Add Field** and set the parameters as required.



Parameter	Description
Field Name	The field to be used as a metric. Select a field from the drop-down list. Note The drop-down list displays the fields in the referenced table. Each field can be added only once.
Metric Name	The name of the metric.
Data Type	The data type of the field. Default value: string .
Metric Type	The type of the metric. Valid values: Basic Attributes , Basic Metrics , and My Metrics .
Connectivity/Availability	The connection status of the field. A field can be in the Available or Unavailable state. If a field in the Metric Registration section is in the Unavailable state, you cannot save the added fields.

b. Click **Save**. The fields are added to the metadata set.

- o Delete a field from the metadata set.

Find the field that you want to delete and click the icon in the **Actions** column to delete the field.

Note

- You cannot delete the first field in the **Required Field** section.
- You cannot edit system metadata sets.

3. Click **Save**. The metadata set is modified.

Check a metadata set

On the Metadata Registration page, find the metadata set that you want to check and click the  icon in the **Actions** column to check the metadata set.

Change the owner of a metadata set

1. On the Metadata Registration page, find the metadata set for which you want to change the owner and click the  icon in the **Actions** column. To change the owners of multiple metadata sets at a time, select the metadata sets and click **Change Owner** in the lower part of the page.
2. In the **Change Owner** dialog box, select the user to whom you want to transfer the metadata set.
3. Click **Change Owner**. The metadata set is transferred to the specified user.

Note

- After the owner of a metadata set is changed, the original owner cannot modify the metadata set.
- You can change the owner of a metadata set to any user of the current tenant.
- You cannot change owners of system metadata sets.

Delete a metadata set

1. On the **Metadata Registration** page, find the metadata set that you want to delete, click the  icon, and then select **Delete** in the **Actions** column.
2. In the **Delete Metadata Set** message, click **Confirm**.

 Note You cannot delete system metadata sets.

9.14.4.5.3. Artifact management

This topic describes how to create, edit, and delete artifacts.

Context

The following table describes the limits on operations related to artifact management.

Operation	Limit
Create an artifact	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator and project administrator
Edit an artifact	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator, project administrator, and artifact owner
View an artifact	This operation can be performed by all users.
Enable an artifact	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator, project administrator, and artifact owner

Operation	Limit
Disable an artifact	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator, project administrator, and artifact owner
Change the owner	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator, project administrator, and artifact owner
Delete an artifact	Limits on the role: <ul style="list-style-type: none"> • System built-in artifact: super administrator only • Custom artifact: super administrator, project administrator, and artifact owner

Create an artifact

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the Data Assets page, click **Administration** in the top navigation bar.
4. On the Data Governance page, click **Artifact Management** under Artifact Management in the left-side navigation pane.
5. On the **Artifact Management** page, click **Create Artifact** in the upper-right corner.
6. On the **Create Artifact** page, set the parameters as required.
 - i. Select an artifact type.

Artifact type	Description
Custom artifact	The artifact that you can customize based on your business needs.
System built-in artifact	The artifact that can be created by the super administrator by using an Apsara Stack account. Dataphin pushes notifications for system built-in artifacts to all users.

ii. Set the basic parameters for the artifact.

Parameter	Description
Artifact Name	<p>The name of the artifact. Dataphin has the following limits on the artifact name:</p> <ul style="list-style-type: none"> ▪ The name must be 1 to 20 characters in length. ▪ The name must be unique.
Governance Target	<p>The type of the data object to be managed. Valid values:</p> <ul style="list-style-type: none"> ▪ Physical Table After you select this option, display fields and governance operations are selected by default. You can clear some display fields and operations as required. The following display fields and operations are available: <ul style="list-style-type: none"> ▪ Display fields: Project Name, Business Unit, Owner, Health Score, Storage Space, Retention Period, and Number of Days (Accessed from First Day to Last Day in the Last 33 Days) ▪ Operations: Move to Recycle Bin, Set Retention Period, and Pause Node The selected display fields and operations will appear in the governance console. ▪ Logical Table After you select this option, display fields and governance operations are selected by default. You can clear some display fields and operations as required. The following display fields and operations are available: <ul style="list-style-type: none"> ▪ Display fields: Project Name, Business Unit, Owner, Health Score, Storage Space, and Retention Period ▪ Only the Set Retention Period operation is supported. You cannot clear this operation. The selected display fields and operations will appear in the governance console. ▪ Task Node After you select this option, display fields and governance operations are selected by default. You can clear some display fields and operations as required. The following display fields and operations are available: <ul style="list-style-type: none"> ▪ Display fields: Node ID, Node Type, Owner, Project Name, Health Score, CPU Consumption, and Duration ▪ Operations: Pause Node and Unpublish Node The selected display fields and operations will appear in the governance console.
Governance Scope	<p>The business scope of the governance target. This parameter is automatically set based on the selected governance target. Valid values: Compute and Storage.</p>
Description	<p>The description of the artifact.</p>

iii. Set rules for the artifact.

Section	Parameter	Description
---------	-----------	-------------

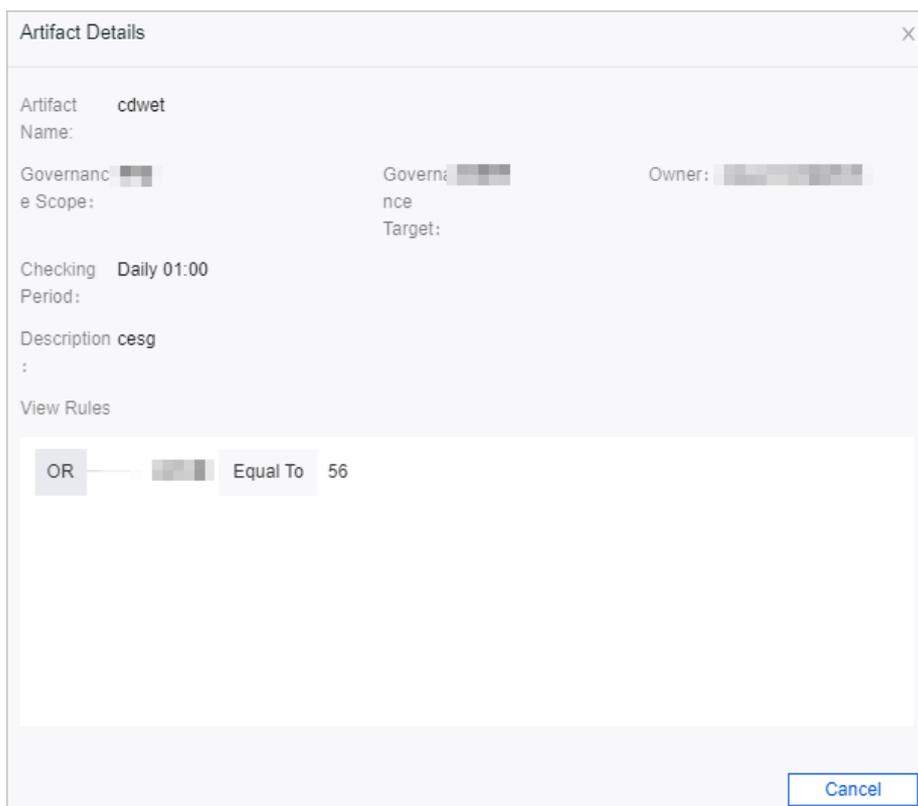
Section	Parameter	Description
Edit Rules	Governance Method	<p>The method of setting rules for the artifact. You can use the codeless UI or code editor.</p> <ul style="list-style-type: none"> If you set the Artifact Type parameter to Custom Artifact, you can select Create Custom Rules or Reference System Rules in the Configure Rules section. If you set the Artifact Type parameter to General Artifact, you can select only Create Custom Rules in the Configure Rules section.
	Code Editor	<p>The custom SQL statements for setting rules for the artifact. If you set the Governance Method parameter to Code Editor, enter SQL statements in the code editor. Example:</p> <pre>select table_guid ,business_name ,is_important from physical_table_individuation where ds='\${bizdate}' and business_name='tmall' and is_relation='Y' and table_size>100</pre> <p> Note This method uses SQL statements to customize rules based on the metrics of registered metadata sets. The artifact uses these rules to manage governance targets.</p>
Configure Rules	Create Custom Rules	<p>To add rules, perform the following steps:</p> <ol style="list-style-type: none"> To add an operator, click Add Operator and select OR or AND. To add a rule, click Add Rule, select a tag and a relational operator, and then enter a value. <p>Parameter description:</p> <ul style="list-style-type: none"> Tags are registered metrics of metadata sets. Relational operators include Greater Than, Greater Than or Equal To, Equal To, Less Than or Equal To, and Less Than. <p>To remove an operator or a rule, click the  icon next to the operator or rule.</p>

Section	Parameter	Description
	Reference System Rules	<p>To reference the rules of a system built-in artifact, perform the following steps:</p> <ol style="list-style-type: none"> Click Reference System Rules. In the Reference System Rules dialog box, select a system built-in artifact from the drop-down list. <div style="border: 1px solid #add8e6; padding: 5px; margin: 10px 0;"> <p>? Note The system displays details of the selected system built-in artifact.</p> </div> <ol style="list-style-type: none"> Click Confirm.

7. Click **Submit**. The artifact is created.

View an artifact

- On the **Artifact Management** page, find the artifact that you want to view and click the  icon in the **Actions** column.
- In the **Artifact Details** message, view the details of the artifact.



Edit an artifact

- On the **Artifact Management** page, find the artifact that you want to edit and click the  icon in the **Actions** column.
- On the **Edit Artifact** page, modify the parameters for the artifact. For more information, see [Create an artifact](#).

Enable an artifact

1. On the **Artifact Management** page, find the artifact that you want to enable and click the  icon in the **Actions** column.
2. In the **Enable Artifact** message, click **Confirm**.

Disable an artifact

1. On the **Artifact Management** page, find the artifact that you want to disable and click the  icon in the **Actions** column.
2. In the **Disable Artifact** message, click **Confirm**.

Change the owner of an artifact

1. On the **Artifact Management** page, find the artifact for which you want to change the owner and click the  icon in the **Actions** column.
2. Select **Change Owner**.
3. In the **Change Owner** dialog box, select the user to whom you want to transfer the artifact.
4. Click **Confirm**.

 **Note** After the artifact is transferred to the selected user, the original owner cannot modify the artifact.

Delete an artifact

1. On the **Artifact Management** page, find the artifact that you want to delete and click the  icon in the **Actions** column.
2. Select **Delete**.
3. In the **Delete Artifact** message, click **Confirm**.

9.14.4.5.4. Push management

Dataphin pushes governance results to you by using SMS messages, emails, and internal messages. This topic describes how to create, edit, and delete push tasks.

Go to the Push Management page

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the Data Assets page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Push Management** under **Artifact Management** in the left-side navigation pane.
5. On the Push Management page, click the **As Owner** or **All** tab to view your push tasks or all push tasks.

Parameter	Description
Topic-based Push Task	The name of the push task.

Parameter	Description
Owner	The owner of the push task.
Start Date	The start date of the push task.
End Date	The end date of the push task.
Status	The status of the push task. Valid values: <ul style="list-style-type: none"> ○ Publish ○ Pause ○ Unpublish

You can enter a keyword in the search box to search for push tasks whose names contain the keyword. You can also click the  icon and set **Status** to filter push tasks.

Create a push task

1. On the **Push Management** page, click **Create Push Task** in the upper-right corner.
2. On the **Create Push Task** page, set the parameters as prompted.
 - i. Set the parameters in the **Push Task Settings** section.

Parameter	Description
Topic-based Push Task	The name of the push task.
Effective Time Range	The period during which the push task is effective.
Notification Time	The time for triggering the push task. <ul style="list-style-type: none"> ■ If you select Specified Time, select a date and the time for pushing governance results. ■ If you select Recurrence, select an interval and the time for pushing governance results.
Notification Methods	The methods for pushing governance results. Valid values: SMS , Email , and Internal Message .

ii. Set the parameters in the **Push Data Settings** section.

Button	Description
Create Artifact	If the artifacts you want are unavailable, click Create Artifact to create ones. For more information, see Create an artifact .
Add Existing Artifacts	<p>To add an existing artifact, perform the following steps:</p> <ol style="list-style-type: none"> Click Add Existing Artifacts. In the Add Existing Artifacts dialog box, find the artifact that you want to add in the Available Artifacts section and click the  icon to add it to the Selected Artifacts section. You can filter artifacts by using one of the following methods: <ul style="list-style-type: none"> Enter a keyword in the search box to search for artifacts whose names contain the keyword. Click the  icon and set Type, Owner, and Governance Target to filter artifacts. Click OK.

iii. Click **Next** in the **Push Data Settings** section.

iv. Select a project and click **Test**.

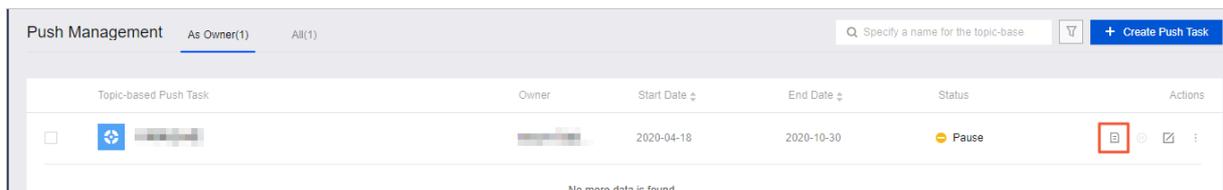
3. Wait until the **Sent** message appears on the **Create Push Task** page, and click **Submit**.

4. In the **Submit Confirm** message, click **Confirm Submit**.

 **Note** By default, a new push task enters the **Publish** state after you submit it.

View a push task

On the **Push Management** page, find the push task that you want to view and click the  icon in the **Actions** column. In the **View Details** message, you can view the settings and involved artifacts and projects about the push task.



Edit a push task

1. On the **Push Management** page, find the push task that you want to edit and click the  icon in the **Actions** column.

2. On the **Edit Push Task** page, set the parameters as prompted. For more information, see [Create a push task](#).

3. Wait until the **Sent** message appears on the **Edit Push Task** page, and click **Submit**.

4. In the **Submit Confirm** message, click **Confirm Submit**.

Pause a push task

1. On the **Push Management** page, find the push task that you want to pause and click the  icon in the **Actions** column.
2. In the **Pause Confirm** message, click **Confirm Pause**.

Change the owner of a push task

1. On the **Push Management** page, find the push task for which you want to change the owner, click the  icon, and then select **Change Owner** in the **Actions** column. You can also select multiple push tasks and click **Change Owner** in the lower part of the page. In the **Change Owner** dialog box, select the user to whom you want to change the owner of the push tasks.
2. In the **Change Owner** dialog box, select the user to whom you want to transfer the push task.
3. Click **Confirm**.

Note

- After the owner of a push task is changed, the original owner cannot modify the push task.
- You can transfer a push task to any user of the current tenant.

Delete a push task

1. On the **Push Management** page, find the push task that you want to delete, click the  icon, and then select **Delete** in the **Actions** column.
2. In the **Delete Confirm** message, click **Confirm Delete**.

9.14.4.5.5. Task management

On the **Task Management** page, you can manage artifact tasks and push tasks. This topic describes how to view task details and task logs and rerun tasks.

Artifact tasks are used as examples in this topic. Follow the same procedures to manage push tasks.

Go to the Task Management page

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Task Management** under **Artifact Management** in the left-side navigation pane.
5. On the **Task Management** page, click the **Push** or **Artifact** tab to view push tasks or artifact tasks.

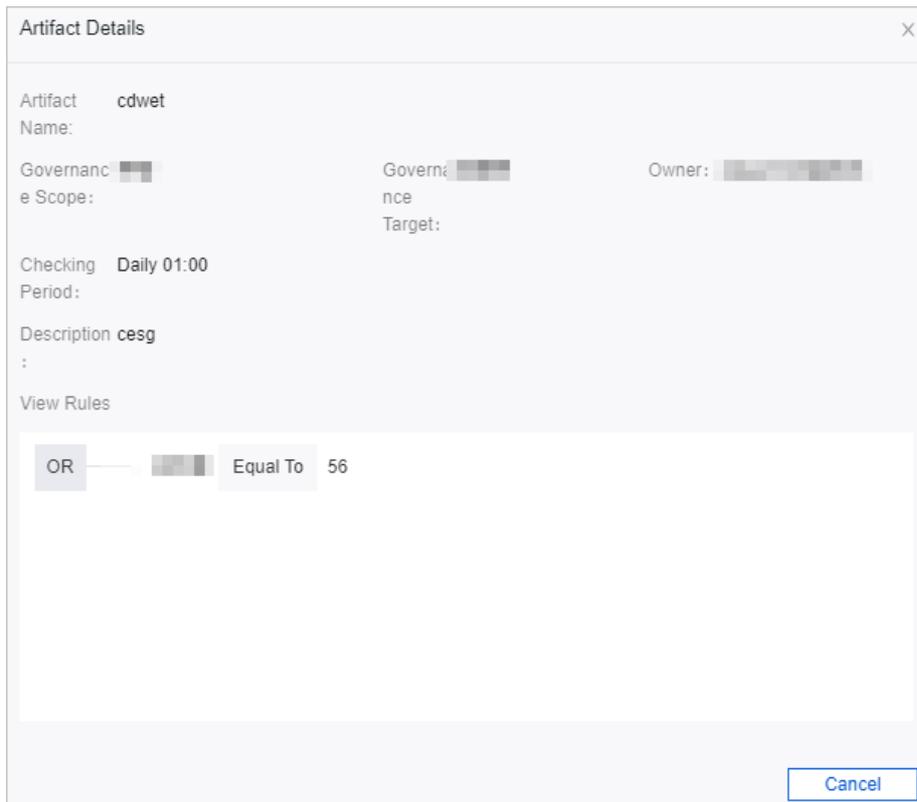
Parameter	Description
Task Name	The name of the task.
Owner	The owner of the task.

Parameter	Description
Task Status	The running status of the task. Valid values: <ul style="list-style-type: none"> ◦ Succeeded ◦ Failed ◦ Initializing ◦ Executing ◦ Stopped
Date Timestamp	The date on which the task is scheduled. The value of this parameter is the default value of the bizdate parameter.
Runtime	The period of time during which the task was run.

You can enter a keyword in the search box to search for tasks whose names contain the keyword. You can also click the  icon and set **Owner**, **Date Timestamp**, **Runtime**, and **Task Status** to filter tasks.

View the details of a task

On the Task Management page, find the task that you want to view and click the  icon in the **Actions** column. The View Details message appears.



Rerun a task

On the **Task Management** page, find the task that you want to rerun and click the  icon in the **Actions** column to rerun the task. Then, the task enters the **Executing** state. After the task is rerun, the task enters the **Succeeded** state.

Pause a task

On the **Task Management** page, find the task that you want to pause and click the  icon in the **Actions** column to pause the task.

View the logs of a task

On the **Task Management** page, find the task whose logs you want to view, click the  icon, and then select **View Log** in the **Actions** column. The View Task Log message appears.

9.14.4.6. Recycle bin

To prevent misoperations on tables, Dataphin allows you to store deleted tables in the recycle bin for a short period of time. If a table is deleted by mistake, you can restore the table from the recycle bin. If a table is no longer required, you can clear the table from the recycle bin. This topic describes how to clear and restore tables in the recycle bin and view the operational logs of a table.

Go to the Tables page

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the Data Assets page, click **Administration** in the top navigation bar.
4. On the **Data Governance** page, click **Tables** under Recycle Bin in the left-side navigation pane.
5. On the **Tables** page, click the **To Be Processed**, **Processed**, or **All** tab to view pending tables, processed tables, or all tables that are moved to the recycle bin.

Parameter	Description
Table Name	The name of the table.
Project Name	The project to which the table belongs.
Table Type	The type of the table. Valid values: Physical Table and Logical Table.
Moved to Recycle Bin At	The date and time when the table is moved to the recycle bin, that is, the date and time when you delete the table in the governance console.
Processed At	The date and time when the table is processed in the recycle bin.
Status	The status of the table.

You can enter a keyword in the search box to search for tables whose names contain the keyword. You can also click the  icon next to **All** and select a project to filter tables.

Clear a table

Tables cannot be restored after they are cleared from the recycle bin. Exercise caution when you perform this operation.

1. On the **Tables** page, find the table that you want to clear and click the  icon in the **Actions** column.
2. In the **Clear Data** message, click **OK**.

Clear multiple tables

Tables cannot be restored after they are cleared from the recycle bin. Exercise caution when you perform this operation. You can clear multiple tables at a time.

1. On the **Tables** page, select multiple tables to be cleared and click **Clear** in the lower part of the page.
2. In the **Clear Data** message, click **OK**. If the status of the selected tables changes to **Succeeded**, they are cleared from the recycle bin.

Restore a table

You can restore a table to the directory where the table was stored before it is moved to the recycle bin.

1. On the **Tables** page, find the table that you want to restore and click the  icon in the **Actions** column.
2. In the **Restore Data** message, click **OK**.

Restore multiple tables

You can restore multiple tables to the directories where the tables were stored before they are moved to the recycle bin.

1. On the **Tables** page, select multiple tables that you want to restore and click **Restore** in the lower part of the page.
2. In the **Restore Data** message, click **OK**. If the status of the selected tables changes to **Succeeded**, they are restored from the recycle bin.

View the operational logs of a table

1. On the **Tables** page, find the table whose operational logs you want to view and click the  icon in the **Actions** column.
2. In the **View Operations** message, view information about operations on the table, including the user who performed an operation, the time when the operation was performed, and the details of the operation.
3. Click **OK**.

9.14.5. Data quality

9.14.5.1. Overview

The Quality module of Dataphin provides a comprehensive data quality scheme that has various features. For example, you can configure quality rules to check data quality and use intelligent alerting.

Background information

As more demanding requirements for building, managing, and applying big data platforms emerge, Dataphin must adapt to more complex scenarios. The raw data from business systems may not meet specific requirements. To properly make business decisions based on data, you can configure quality rules in the Quality module of Dataphin to check the data quality, including the data timeliness, accuracy, integrity, consistency, and validity.

Features

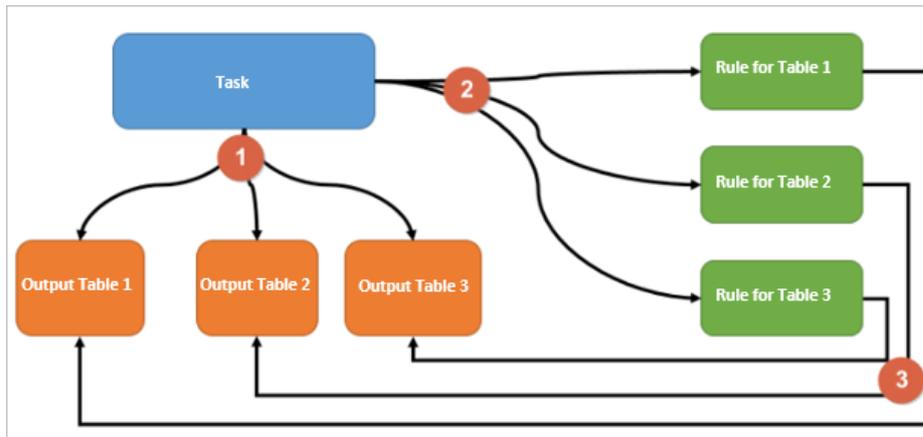
You can use Quality to check the quality of various tables, trigger alerts, and generate quality reports.

The Quality module consists of the following submodules:

- **Quality Overview**: provides an overview of quality rule-based checks.
- **Quality Management**: allows you to configure quality rules. You can also view quality rules, check records,

and quality reports.

The following figure shows the process of quality rule-based check.



9.14.5.2. Terms

This topic introduces the basic terms that are involved in the Quality module, such as the data object, quality rule, strong rule, and weak rule.

Term	Description
data object	An object that is managed in Quality. A data object can be a logical table or a physical table.
quality rule	A rule that is used to check the quality of a data object. A quality rule can check data quality at the granularity of fields. For example, you can configure a rule to monitor changes in the number of rows in a table at the ODS layer and check whether duplicate primary keys exist in the table.
strong rule	A rule whose check results affect the running of downstream nodes. If a data object fails a strong rule-based check, the system does not schedule downstream nodes of the current node where the data object resides. In addition, the system sets the status of the current node to failed and reports an alert to the specific alert recipients.
weak rule	A rule whose check results do not affect the running of downstream nodes. If a data object fails a weak rule-based check, the system does not forcibly stop scheduling downstream nodes of the current node, but only reports an alert to the specific alert recipients.
check record	A record that Quality generates after the quality of a data object is checked based on quality rules.
quality report	A report that Quality generates after the quality of a data object is checked based on quality rules. After the node where a data object resides is scheduled, Quality checks the quality of the updated data object based on the configured quality rules. Then, Quality generates a quality report that contains detailed results of the quality rule-based check.
partition expression	An expression that is used to define the data range to which a quality rule is applied.

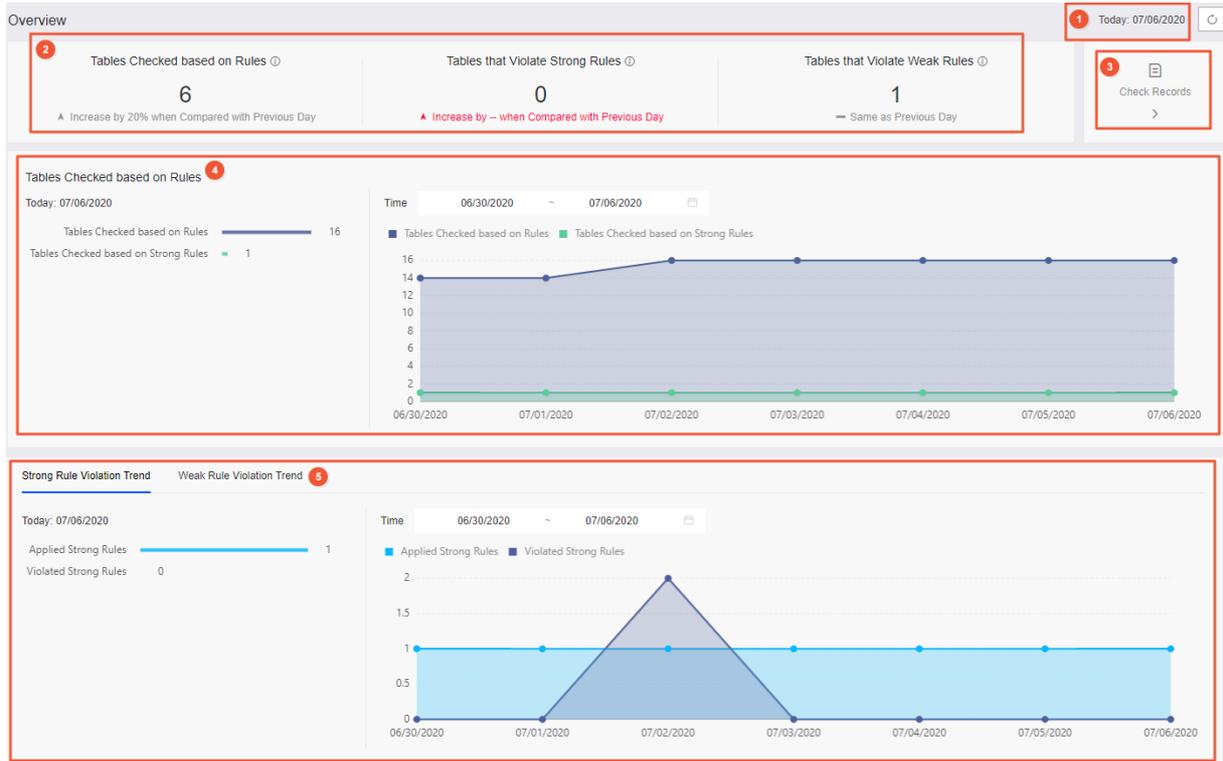
9.14.5.3. View statistics on the Overview page

The Overview page provides the overall results of quality rule-based checks on tables. It helps you properly identify and process abnormal check results.

To go to the **Overview** page, perform the following steps:

1. Log on to the [Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.

On the **Overview** page, you can view the overall results of quality rule-based checks on tables.



No.	Description
1	Displays the date of the current day, which is used as the timestamp of data displayed on the Overview page.
2	<ul style="list-style-type: none"> • Displays the change in the number of tables that are checked based on quality rules on the current day, compared with that on the previous day. • Displays the change in the number of tables that fail strong rule-based checks on the current day, compared with that on the previous day. • Displays the change in the number of tables that fail weak rule-based checks on the current day, compared with that on the previous day.
3	Provides an entry for you to go to the Check Records page.

No.	Description
4	<p>Displays the following data by default:</p> <ul style="list-style-type: none"> • The number of tables that are checked based on quality rules on the current day and the trend within the latest seven days. • The number of tables that are checked based on strong rules on the current day and the trend within the latest seven days. <p>Allows you to specify a time range within the last 30 days to view corresponding data.</p>
5	<ul style="list-style-type: none"> • Displays the trend of statistics on strong rules, including: <ul style="list-style-type: none"> ◦ The number of strong rules that are used to check tables on the current day and the trend within the latest seven days. ◦ The number of strong rules that tables violate on the current day and the trend within the latest seven days. • Displays the trend of statistics on weak rules, including: <ul style="list-style-type: none"> ◦ The number of weak rules that are used to check tables on the current day and the trend within the latest seven days. ◦ The number of weak rules that tables violate on the current day and the trend within the latest seven days. <p>Allows you to specify a time range within the last 30 days to view corresponding data.</p>

9.14.5.4. Manage quality rules

9.14.5.4.1. View quality rules

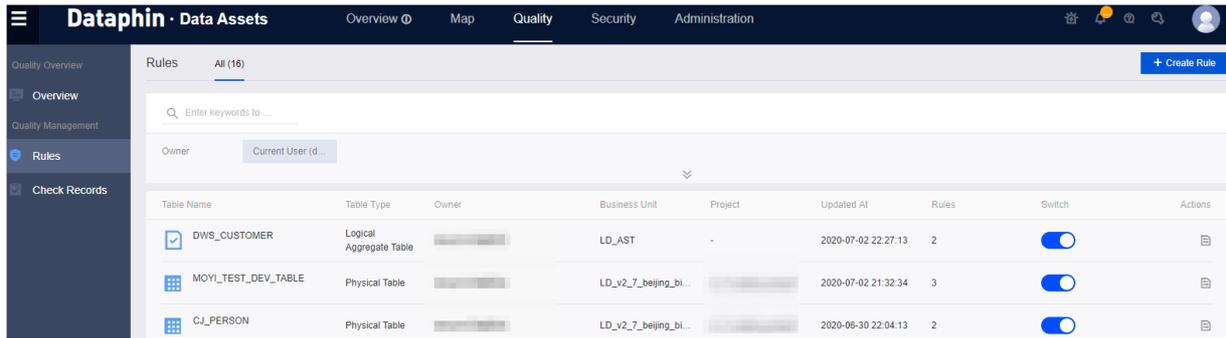
The Rules page displays information about quality rules that are configured for tables. This topic describes how to view quality rules and enable or disable quality rule-based check for tables.

Go to the Rules page

To go to the Rules page, perform the following steps:

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the Data Assets page, click **Quality** in the top navigation bar.
4. On the Quality page, click **Rules** in the left-side navigation pane.

On the Rules page, you can view information about quality rules that are configured for tables.



You can set **Owner**, **Business Unit**, **Project**, and **Table Type** to filter records. You can also enter a keyword in the search box to search for records of tables whose names contain the keyword.

Parameter	Description
Table Name	The name of the table for which quality rules are created.
Table Type	The type of the table. Valid values: <ul style="list-style-type: none"> • Logical Aggregate Table • Logical Dimension Table • Logical Fact Table • Physical Table
Owner	The owner of the table.
Business Unit	The business unit to which the table belongs. If the project to which a physical table belongs is not bound to a business unit, a hyphen (-) appears in the Business Unit column of the physical table.
Project	The project to which the table belongs.
Updated At	The last time when the quality rules of the table were updated.
Rules	The number of quality rules that are configured for the table.
Switch	Specifies whether quality rule-based check is enabled for the table. <ul style="list-style-type: none"> • <input checked="" type="checkbox"/> indicates that quality rule-based check is enabled for the table. • <input type="checkbox"/> indicates that quality rule-based check is disabled for the table.
Actions	The operations that you can perform on the table. You can click the  icon in the Actions column to go to the Rule Settings page of the table.

View the settings of a rule

On the **Rules** page, find the table whose settings you want to view and click the  icon in the **Actions** column. On the **Rule Settings** page, you can view the rule settings of the table.

Enable or disable quality rule-based check for a table

On the **Rules** page, find the table for which you want to disable or enable quality rule-based check and click the  or  icon in the **Switch** column.

9.14.5.4.2. Configure a quality rule

To properly manage data quality, you must configure quality rules. This topic describes how to configure a quality rule for a table.

Prerequisites

Tables are submitted or published to the production environment.

Step 1: Create a quality rule

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.
4. On the **Quality** page, click **Rules** in the left-side navigation pane.
5. On the **Rules** page, click **Create Rule** in the upper-right corner.
6. In the **Create Rule** dialog box, select a table and click **Configure**. You can select a table by using one of the following methods:
 - Click the  icon and set **Business Unit**, **Project**, and **Table Type** to filter tables. Then, select the desired table from the drop-down list.
 - Click the arrow button. Enter a keyword in the search box to search for tables whose names contain the keyword and select the desired table.

Step 2: Configure alert information

After you configure alert information for a table, Alert Center reports alerts to the specified recipients if the table fails quality rule-based checks.

1. On the Rule Settings page of the table, click the  icon in the **Alert Settings** section.
2. In the **Alert Settings** dialog box, set **Recipients** and **Methods** and click **OK**.

Step 3: Create a partition expression

A partition expression specifies the data range to which a group of quality rules are applied. You can create multiple partition expressions for a table and configure multiple quality rules under each partition expression, as shown in the following figure.

1. On the Rule Settings page of the table, click **Create Partition Expression** in the **Rule Settings** section.
2. In the **Create Partition Expression** dialog box, set **Type**, enter a partition expression, and then click **Save**.
 - If you set **Type** to a partition expression type that is built in Quality, the system automatically fills in the corresponding partition expression. The following table lists the partition expression types that are built in Quality and the corresponding partition expressions.

Partition expression type	Partition expression
Last Day	ds='\${yyyyMMdd}'

Partition expression type	Partition expression
First Day of the Current Week	<code>ds=tdBeginDate('\${yyyyMMdd}', 'w', 'yyyyMMdd')</code>
Last Day of the Current Week	<code>ds=customEndDate('\${yyyyMMdd}', 'w', '1', 0, 'yyyyMMdd')</code>
First Day of the Current Month	<code>ds=tdBeginDate('\${yyyyMMdd}', 'm', 'yyyyMMdd')</code>
Last Day of the Current Month	<code>ds=customEndDate('\${yyyyMMdd}', 'm', '01', 0, 'yyyyMMdd')</code>
First Day of the Current Quarter	<code>ds=tdBeginDate('\${yyyyMMdd}', 'q', 'yyyyMMdd')</code>
Last Day of the Current Quarter	<code>ds=customEndDate('\${yyyyMMdd}', 'q', '0101', 0, 'yyyyMMdd')</code>
First Day of the Current Year	<code>ds=tdBeginDate('\${yyyyMMdd}', 'y', 'yyyyMMdd')</code>
Last Day of the Current Year	<code>ds=customEndDate('\${yyyyMMdd}', 'y', '0101', 0, 'yyyyMMdd')</code>
First Day of the Previous Month	<code>ds=cBeginDate('\${yyyyMMdd}', 'm', 'yyyyMMdd')</code>
Last Day of the Previous Month	<code>ds=cEndDate('\${yyyyMMdd}', 'm', 'yyyyMMdd')</code>
All Partitions	<code>ds=ALL</code>

- If you set Type to **Custom**, you must enter a partition expression, for example, `ds=${yyyyMM01}`. After you create a partition expression, you can perform the following operations on it:
 - Click the  icon next to the partition expression to edit it.
 - Click the  icon next to the partition expression to delete it.

Step 4: Configure a quality rule

To configure a quality rule under a partition expression, perform the following steps:

1. On the Rule Settings page of the table, click **Create Rule** next to the partition expression.
2. In the **Create Rule** dialog box, set the parameters as prompted and click **Save**. The valid values of the Rule Type parameter are **Template-based Rules** and **Custom Rules**.
 - If you set Rule Type to **Template-based Rules**, you must set other parameters as described in the following table.

Parameter	Description
-----------	-------------

Parameter	Description
Object Name	The object to be checked based on the quality rule. The value of the Object Name parameter is in the Table: Name of the current table or Field: Name of a field in the current table format.
Strong or Weak Rule	The type of the quality rule. You can select Strong Rule or Weak Rule. <ul style="list-style-type: none"> ▪ If you select Strong Rule, Alert Center reports an alert and the system blocks downstream nodes of the current node where the table resides when the specific object fails the rule-based check. ▪ If you select Weak Rule, Alert Center reports an alert but the system does not block downstream nodes of the current node when the specific object fails the rule-based check.
Template	The check dimension of the quality rule. The valid values of the Template parameter vary based on the value you selected for the Object Name parameter. For more information, see Configuration table .
Trend	<ul style="list-style-type: none"> ▪ If the check type is fluctuation check, the valid values of the Trend parameter are Absolute Value, Upward, and Downward. ▪ If the check type is comparison, the value of the Trend parameter is fixed to Fixed Value. For more information, see Configuration table .
Comparison Settings	The method for comparing the rule-based check result with the target value. You must specify a target value if the check type is comparison. The valid values of the Comparison Method parameter are Greater Than Target Value , Greater Than or Equal To Target Value , Equal To Target Value , Less Than Target Value , and Less Than or Equal To Target Value .
Fluctuation Threshold	The fluctuation threshold for the rule-based check result. The value of the Threshold parameter can range from 0.0 to 10.0.

Configuration table

Object name	Template	Trend	Configuration item	Check type
Table: Name of the current table	Compare the Target Table or Partition Size with that of Previous Day	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Number of Table or Partition Rows with that of Previous Day	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Table or Partition Size with that of 7 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check

Object name	Template	Trend	Configuration item	Check type
Table: Name of the current table	Compare the Target Number of Table or Partition Rows with that of 7 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Number of Table or Partition Rows with that of 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Number of Table or Partition Rows with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Average Number of Table or Partition Rows with that of 7 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Average Number of Table or Partition Rows with that of 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Number of Table Partitions	Fixed Value	Comparison Settings	Comparison
Table: Name of the current table	Check Fluctuations in the Number of Table Partitions	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Table: Name of the current table	Compare the Target Number of Table or Partition Rows with that of Previous Day, the First Day of this Month, and that of 7, 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target and the Expected Number of Unique Values	Fixed Value	Comparison Settings	Comparison

Object name	Template	Trend	Configuration item	Check type
Field: Name of a field in the current table	Compare the Target Number of Unique Field Values with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Average Field Value with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Maximum Field Value with that of Previous Day	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Maximum Field Value with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Minimum Field Value with that of Previous Day	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Minimum Field Value with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Sum of Field Values with that of Previous Day	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check
Field: Name of a field in the current table	Compare the Target Sum of Field Values with that of Previous Day and that of 7 and 30 Days Ago	Absolute Value, Upward, and Downward	Fluctuation Threshold	Fluctuation check

Object name	Template	Trend	Configuration item	Check type
Field: Name of a field in the current table	Number of Null Field Values	Fixed Value	Comparison Settings	Comparison
Field: Name of a field in the current table	Rate of Null Values (Number of Null Values/Total Number of Rows)	Fixed Value	Comparison Settings	Comparison
Field: Name of a field in the current table	Number of Duplicate Field Values	Fixed Value	Comparison Settings	Comparison
Field: Name of a field in the current table	Rate of Duplicate Field Values (Number of Duplicate Values/Total Number of Rows)	Fixed Value	Comparison Settings	Comparison

- o If you set Rule Type to **Custom Rules**, you must set other parameters as described in the following table.

Parameter	Description
Rule Name	The name of the quality rule.
Details	<p>The content of the quality rule. Example:</p> <pre>select sum(value) as metric from current_table ctb left outer join related_table rtb on ctb.id = rtb.id where ds = \${bizdate};</pre>
Strong or Weak Rule	<p>The type of the quality rule. You can select Strong Rule or Weak Rule.</p> <ul style="list-style-type: none"> ▪ If you select Strong Rule, Alert Center reports an alert and the system blocks downstream nodes of the current node where the table resides when the specific object fails the rule-based check. ▪ If you select Weak Rule, Alert Center reports an alert but the system does not block downstream nodes of the current node when the specific object fails the rule-based check.
Check Type	<p>The check type of the quality rule. The valid values of the Check Type parameter are Compare Target Data with that of Previous Day, Compare Target Data with that of 7 Days Ago, Compare the Target Average Value with that of 7 Days Ago, Compare the Target Average Value with that of 30 Days Ago, and Compare with Fixed Value.</p>

Parameter	Description
Trend	<ul style="list-style-type: none"> ▪ If the check type is fluctuation check, the valid values of the Trend parameter are Absolute Value, Upward, and Downward. ▪ If the check type is comparison, the value of the Trend parameter is fixed to Fixed Value.
Comparison Settings	The method for comparing the rule-based check result with the target value. You must specify a target value if the check type is comparison. The valid values of the Comparison Method parameter are Greater Than Target Value , Greater Than or Equal To Target Value , Equal To Target Value , Less Than Target Value , and Less Than or Equal To Target Value .
Fluctuation Threshold	The fluctuation threshold for the rule-based check result. The value of the Threshold parameter can range from 0.0 to 10.0.

3. Select the configured quality rule in the rule list and click **Test Run** in the lower part of the list. After the test is successful, turn on the switch for the quality rule. Then, the quality rule takes effect. The test run can be in one of the following states:
 - **Pending**
 - **Succeeded**
 - **Failed**
 - **Running**

9.14.5.4.3. Modify a quality rule

To properly manage data quality, you must configure quality rules. This topic describes how to modify a quality rule for a table.

Prerequisites

A quality rule is configured for a table. For more information, see [Configure a quality rule](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.
4. On the **Quality** page, click **Rules** in the left-side navigation pane.
5. On the **Rules** page, find the table for which you want to modify a quality rule and click the  icon in the **Actions** column.
6. On the **Rule Settings** page, find the quality rule that you want to modify and click the  icon in the **Actions** column.
7. In the **Change Rule** dialog box, modify the parameter settings as prompted. For more information, see [Configure a quality rule](#).
8. Click **Save**.

9.14.5.4.4. Delete a quality rule

To properly manage data quality, you must configure quality rules. This topic describes how to delete a quality rule for a table.

Prerequisites

A quality rule is configured for a table. For more information, see [Configure a quality rule](#).

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.
4. On the **Quality** page, click **Rules** in the left-side navigation pane.
5. On the **Rules** page, find the table for which you want to delete a quality rule and click the  icon in the **Actions** column.
6. On the **Rule Settings** page, find the quality rule that you want to delete and click the  icon in the **Actions** column.
7. In the **Delete Rule** message, click **OK**.

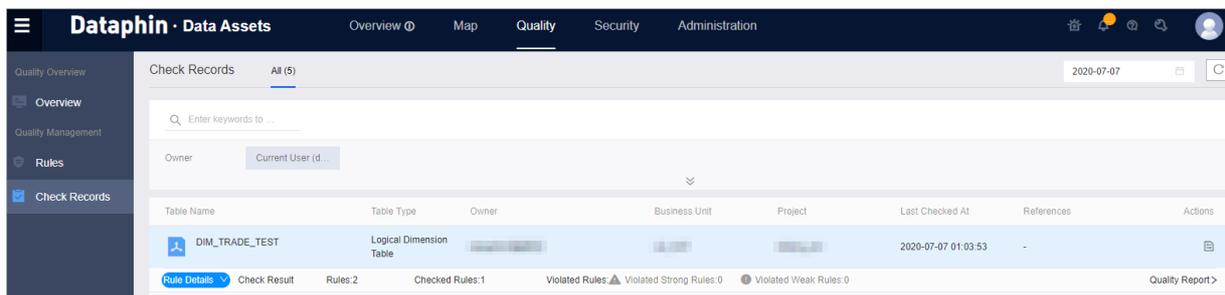
9.14.5.5. View check records

The Check Records page displays records of quality rule-based checks on tables. This topic describes how to view such records.

To go to the **Check Records** page, perform the following steps:

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.
4. On the **Quality** page, click **Check Records** in the left-side navigation pane.

On the Check Records page, you can click the  icon in the upper-right corner to select a date. Then, check records of that day appear on the page.



You can set **Owner**, **Business Unit**, **Project**, **Status**, **Table Type**, and **Rule Template** to filter check records. You can also enter a keyword in the search box to search for check records of tables whose names contain the keyword.

GUI element	Description
Table Name	The name of the table that Quality checked based on quality rules.
Table Type	The type of the table.

GUI element	Description
Owner	The owner of the table.
Business Unit	The business unit to which the table belongs.
Project	The project to which the table belongs.
Last Checked At	The last time when the table was checked based on quality rules.
References	The number of times that the table was referenced by tasks when the tasks were scheduled in the production environment.
Actions	<p>The operations that you can perform on the table. You can click the  icon in the Actions column to go to the Rule Settings page of the table, where you can:</p> <ul style="list-style-type: none"> • View existing quality rules of the table. • Create quality rules for the table. For more information, see Configure a quality rule. • Modify existing quality rules of the table. For more information, see Configure a quality rule.
Rule Details	The details about quality rule-based checks on the table. You can click the  icon next to Rule Details to view the details.
Check Result	<p>The summary of quality rule-based checks on the table, including:</p> <ul style="list-style-type: none"> • Rules: the number of quality rules that have been configured for the table. • Checked Rules: the number of quality rules that Quality used to check the table. • Violated Rules: <ul style="list-style-type: none"> ◦ Violated Strong Rules: the number of strong rules that the table violated. ◦ Violated Weak Rules: the number of weak rules that the table violated.
Quality Report	The quality report of the table. You can click Quality Report to go to the Quality Report page of the table. You can view information such as the quality analysis result, report details, and rule-based check instances on the page. For more information, see View a quality report .

9.14.5.6. View a quality report

After a task that references a table is scheduled in the production environment, Quality checks the quality of the updated table based on the quality rules configured for the table. Then, Quality generates a quality report that contains detailed results of the quality rule-based check. This topic describes how to view a quality report.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Data Assets** in the top navigation bar.
3. On the **Data Assets** page, click **Quality** in the top navigation bar.
4. The **Quality** page appears.
5. Go to the **Quality Report** page of a table by using one of the following methods:

- a. On the Quality page, click **Rules** in the left-side navigation pane.
 - b. On the **Rules** page, find the table whose quality report you want to view and click the  icon in the **Actions** column.
 - c. On the **Rule Settings** page of the table, click **Quality Report**.
 - a. On the Quality page, click **Check Records** in the left-side navigation pane.
 - b. On the **Check Records** page, find the table whose quality report you want to view and click **Quality Report** under the table.
6. On the **Quality Report** page of the table, click the  icon in the upper-right corner to select a date. Quality displays the quality report of the table on that day.

Parameter or section	Section	Parameter	Description
Table Details	N/A	Table	The name of the table.
		Business Unit	The business unit to which the table belongs.
		Project	The project to which the table belongs.
		Table Type	The type of the table. Valid values: <ul style="list-style-type: none"> ○ Logical Aggregate Table ○ Logical Dimension Table ○ Logical Fact Table ○ Physical Table
		Owner	The owner of the table.
		Task Node	The task that referenced the table in the production environment.
		Alert Recipients	The recipients to whom Alert Center reported alerts when the table failed quality rule-based checks.
Quality Analysis Report	N/A	Violated Rules	The number of quality rules that the table violated.
		Checked Rules	The total number of quality rules that are configured for the table and the number of quality rules that Quality used to check the table.
		Instance Execution Time	The time when the task that referenced the table was run in the production environment.
	N/A	Partition Expression	The partition expression of the table.
		Partition	The partition key value that is obtained based on the partition expression, for example, <code>DS=20200212</code> .
			Field Name

Parameter or section	Section	Parameter	Description
Report Details	Field Check	Rule Template Name	The template of the quality rule that is configured for the field.
		Strong or Weak Rule	The type of the quality rule.
		Comparison Method	<p>The method for comparing the quality rule-based check result with the specific fluctuation threshold or target value.</p> <ul style="list-style-type: none"> ◦ If the check type is fluctuation check, the valid values of the Trend parameter are Absolute Value, Upward, and Downward. ◦ If the check type is comparison, the value of the Trend parameter is fixed to Fixed Value.
		Threshold	The fluctuation threshold of the field values.
		Status	The result of the quality rule-based check on the field.
		Details	<p>The details about the field values. To view the value trend of the field, perform the following steps:</p> <ol style="list-style-type: none"> i. Click the  icon in the Details column of the field. ii. In the Alert Trend dialog box, specify a time range in the upper-right corner and view the value trend that appears. iii. Click OK.
	Custom Rule	Custom Rule Name	The name of the custom quality rule.
		Strong or Weak Rule	The type of the custom quality rule.
		Comparison Method	<p>The method for comparing the custom quality rule-based check result with the specific fluctuation threshold or target value.</p> <ul style="list-style-type: none"> ◦ If the check type is fluctuation check, the valid values of the Trend parameter are Absolute Value, Upward, and Downward. ◦ If the check type is comparison, the value of the Trend parameter is fixed to Fixed Value.
		Status	The result of the custom quality rule-based check on the field.

Parameter or section	Section	Parameter	Description
		Details	<p>The details about the field values. To view the value trend of the field, perform the following steps:</p> <ol style="list-style-type: none"> i. Click the  icon in the Details column of the field. ii. In the Alert Trend dialog box, specify a time range in the upper-right corner and view the value trend that appears. iii. Click OK.
	Rule-based Check Instances	N/A	Table Name
Partition			The partition that Quality checked based on the quality rule.
Rule			The quality rule that Quality used to check the partition.
Owner			The owner of the table.
Date Timestamp			The data timestamp that Quality specified based on the partition.
Execution Time (Range)			The duration of the quality rule-based check.
Status			<p>The result of the quality rule-based check. Valid values:</p> <ul style="list-style-type: none"> ◦ Failed ◦ Running ◦ Succeeded
Log			<p>The operational logs of the quality rule-based check instance. You can click the  icon in the Log column to view the operational logs of the quality rule-based check instance.</p>

9.15. Theme-based data service

9.15.1. Ad hoc query

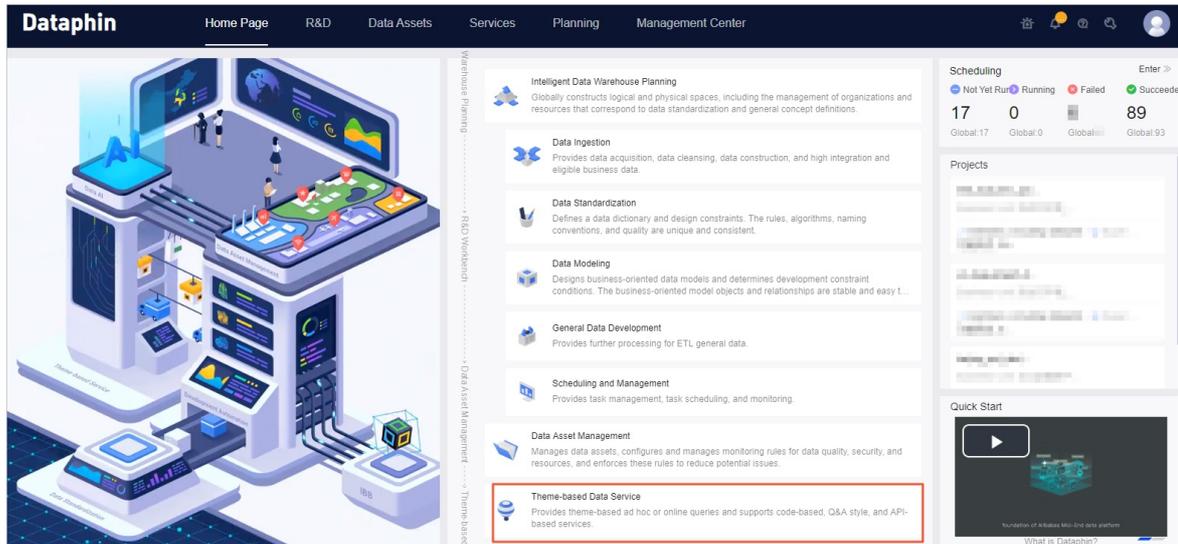
Dataphin provides the ad hoc query feature for you to query data.

Context

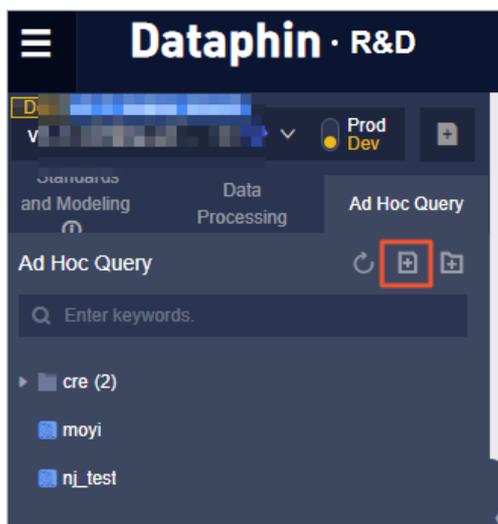
If your computing engine type is Hadoop, the Hive SQL syntax is supported in ad hoc queries. If your computing engine type is MaxCompute, the MaxCompute SQL syntax is supported in ad hoc queries. Dataphin can automatically identify and change the SQL syntax based on your computing engine type.

Procedure

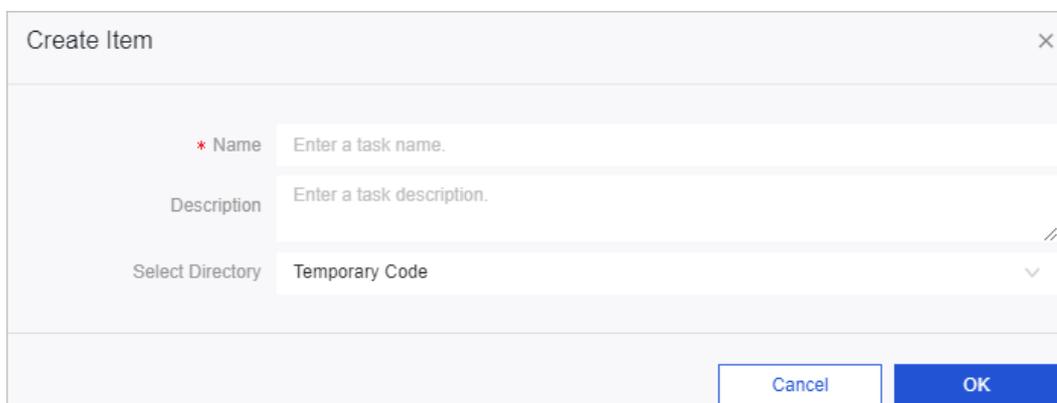
1. Log on to the Dataphin console.
2. On the Dataphin homepage, click **Theme-based Data Service** to go to the Ad Hoc Query tab.



3. On the Ad Hoc Query tab of the Develop tab, click the **Create Item** icon in the left-side navigation pane.

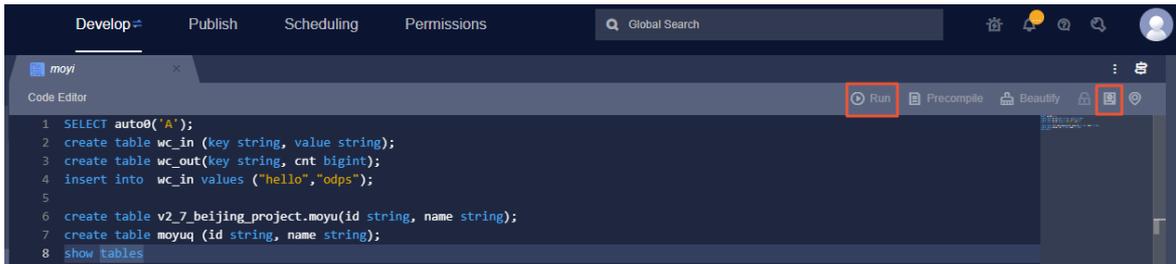


4. In the **Create Item** dialog box that appears, enter the task name and description, select a directory, and then click OK.



5. In the left-side ad hoc query task list, click the ad hoc query task that you created in the preceding step to go to the **Code Editor** tab.

6. Write SQL statements, click the **Save** icon, and then click **Run** in the upper-right corner.



Note When you write SQL statements, reference table names in the format of *Project name.Table name*, for example, *demo.dim_qwe*.

7. After SQL statements are run, view the results on the **Result** tab.

9.16. Data services

9.16.1. Overview

As the last step in constructing a Data Mid-End based on Dataphin, the Services module provides data services in a uniform manner. In the Services module, Dataphin centrally manages data services. This effectively facilitates and secures the use of data services.

Dataphin is designed to build a uniform data service bus to help enterprises increase the value of their data assets while guaranteeing data reliability, security, and effectiveness. The Services module allows you to create API operations in the easy-to-use template wizard mode. This helps enterprises maximize the value of data applications that they create.

The Services module of Dataphin consists of the **API Service**, **Workbench**, and **Administration** pages.

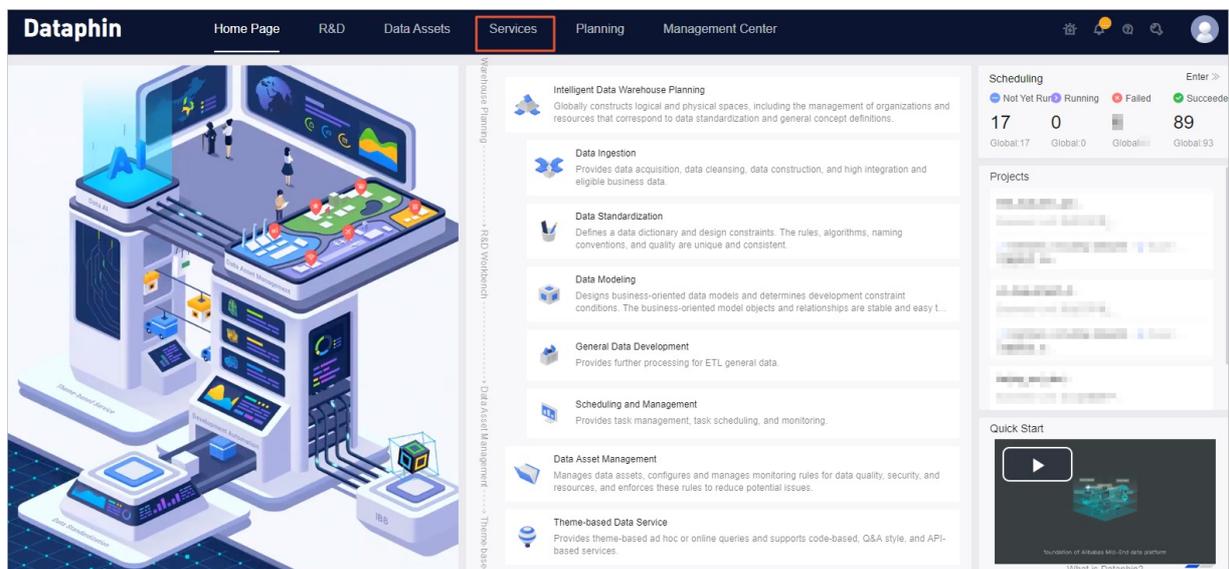
- The **API Service** page lists all published API operations.
- The **Workbench** page is composed of **Development Center** and **Application Center**.
 - **Development Center** is designed for data service providers to develop services. It contains two modules: **Service Units** and **APIs**.
 - **Application Center** is designed for data service users to consume API services from applications. It contains two modules: **APIs** and **Applications**.
- The **Administration** page contains the following modules: **Member Management**, **Groups**, **Metadata Management**, **Monitoring**, **Network Configuration**, Call Examples, and Dataphin Data Sources. On this page, you can monitor and manage data services in a uniform manner.

In the Services module, you can develop, call, and monitor API operations.

Feature	Procedure
---------	-----------

Feature	Procedure
Develop API operations	<ol style="list-style-type: none"> 1. Add a Dataphin user to the Services module as a developer. For more information, see Add a member. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p>Note Only a developer can create API operations in the Services module.</p> </div> <ol style="list-style-type: none"> 2. Create a group. For more information, see Create a group. 3. Create a service unit. For more information, see Create a metadata set. 4. Create an API operation. For more information, see Create an API operation in template wizard mode. 5. Test the created API operation. For more information, see Test an API operation. 6. Publish the API operation. For more information, see Publish an API operation.
Call API operations	<ol style="list-style-type: none"> 1. Add a Dataphin user to the Services module as an application user. For more information, see Add a member. <div style="border: 1px solid #ccc; background-color: #e6f2ff; padding: 10px; margin: 10px 0;"> <p>Note Only an application user can call API operations in the Services module.</p> </div> <ol style="list-style-type: none"> 2. Create an application. For more information, see Create an application. 3. Apply for the permission to use an API operation. For more information, see Apply for permissions on an API operation. 4. Debug the requested API operation. For more information, see Debug an API operation. 5. Call the API operation. For more information, see Call an API operation.
Monitor API operations	<p>For more information about how to monitor API operations, see Monitor an API operation.</p>

Log on to the [Log on to the Dataphin console](#). On the Dataphin homepage, click **Services** in the top navigation bar. The Service page appears.



9.16.2. Terms

This topic describes the terms of the Services module in Dataphin, such as Dataphin data source, API operation, group, service unit, and application.

Term	Description
Dataphin data source	The data source that is created based on a table.
API operation	The API operation that is created by configuring a table.
Service unit	Allows you to manage the metadata of one or more tables in a uniform manner.
Application	The permission subject that allows members to consume data services, including calling API operations and using Dataphin data sources.
Group	Allows you to classify and manage data services and applications based on specific features or scenarios.
API Service page	The page that displays the API operations that you have published. It is the API market of Dataphin.
Metadata management	Allows you to configure and manage the schema mapping from NoSQL databases to relational databases to guarantee the consistency of data sources.
Network configuration	The network configuration for an application to access API operations and data sources. Network configuration includes the domain name configuration and the configuration of the Virtual Private Cloud (VPC) whitelist.
Call Examples page	The page that provides the instructions for calling API operations and using Dataphin data sources.

9.16.3. Configure the network

Before an application can call API operations and use Dataphin data sources, you must configure the network. Network configuration includes the domain name configuration and the configuration of the VPC whitelist. This topic describes how to enable and disable a second-level domain name, configure an independent domain name, and unbind an independent domain name to the current Apsara Stack tenant account.

Context

You can configure the network only by using an Apsara Stack tenant account.

Enable and disable a second-level domain name

This section describes how to enable and disable the second-level domain name on the Internet. You can follow the same procedure to enable and disable the second-level domain name in a VPC.

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Services** in the top navigation bar.
3. On the Service page, click **Administration** in the top navigation bar.
4. On the Administration page, click **Network Configuration** in the left-side navigation pane.
5. Disable the second-level domain name.
 - i. Click **Close the public domain second-level domain name** next to the domain name.

- ii. In the **Prompt** message, click **Confirm**.
6. Enable the second-level domain name.
- i. Click **Open the second-level domain name of the public network**.
 - ii. In the **Prompt** message, click **Confirm**.

Configure an independent domain name

1. Bind an independent domain name to the current Apsara Stack tenant account.
 - i. On the **Domain Configuration** tab, click **Bind Domain Name** in the upper-right corner of the **Independent Domain Name** section.
 - ii. In the **Bind Domain Name** dialog box, enter a domain name.
 - iii. Click **Confirm** to bind the independent domain name to the current Apsara Stack tenant account.
2. Bind an SSL certificate to the independent domain name.
 - i. Click **Add** in the **SSL Certificate** column.
 - ii. In the **New Certificate** dialog box, enter a certificate name, the certificate content, and the private key as required.

New Certificate
✕

* Certificate Name:

* Certificate Content:

Example:
 -----BEGIN CERTIFICATE-----
 MIIFtDCCBJygAwIBAgIQRgWF1j00cozRI1pZ+ultKTANBgkqhkiG9w0BAQsFADBP
 ...
 -----END CERTIFICATE-----

* Private Key:

Example:
 -----BEGIN RSA PRIVATE KEY-----
 MIIEpAIBAAKCAQEAA8GjlleJ7rlo86mtbcDnUfqzTQAm4b3zZEo1aKsfAuwcvCud

 -----END RSA PRIVATE KEY-----

Cancel Confirm

Parameter	Description
Certificate Name	The name of the SSL certificate.
Certificate Content	The content of the SSL certificate. Enter the content based on the example on the page.
Private Key	The private key of the SSL certificate. Enter the private key based on the example on the page.

- iii. Click **Confirm** to bind the SSL certificate.

Unbind an independent domain name

1. In the **Independent Domain Name** section, find the independent domain name that you want to unbind and click the  icon in the **Actions** column.

2. In the message that appears, click **OK**.

9.16.4. Add a member

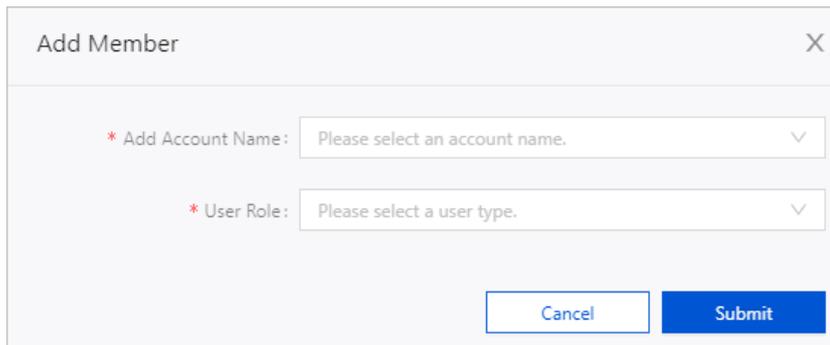
The Services module allows you to centrally manage the members who develop and use data services in Dataphin. This topic describes how to add a member to the Services module.

Context

A Dataphin user can access the Services module only after the user is added as a member with the **Developer** or **Application User** role. Only members can develop and use API operations and data sources.

Procedure

1. [Log on to the Dataphin console](#).
2. On the Dataphin homepage, click **Services** in the top navigation bar.
3. On the Service page, click **Administration** in the top navigation bar.
4. On the Administration page, click **Member Management** in the left-side navigation pane.
5. On the **Member Management** page, click **Add Member** in the upper-right corner.
6. In the **Add Member** dialog box, select the name of the account to be added as a member and the user type from the **Add Account Name** and **User Role** drop-down lists, respectively.



Parameter	Description
Add Account Name	The name of the account to be added as a member of the Services module.
User Role	The role to be assigned to each selected account. Valid values: <ul style="list-style-type: none">◦ Developer: Developers provide data services and develop API operations.◦ Application User: Application users use data services and call API operations.

7. Click **Submit**.

The following table describes the operations that you can perform on the added members.

Operation	Description
Change the user role of a member	Click the  icon in the Actions column to change the user role of a member.

Operation	Description
Delete a member	<ul style="list-style-type: none"> Click the  icon in the Actions column to delete a member from the Services module. To delete multiple members at a time, select the members and click Remove in the upper-right corner of the Member Management page.

9.16.5. Develop API operations

9.16.5.1. Create a group

Dataphin allows you to manage service units and applications by group. This topic describes how to create a group for managing service units and applications.

Procedure

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Services** in the top navigation bar.
3. On the Service page, click **Administration** in the top navigation bar.
4. On the Administration page, click **Groups** in the left-side navigation pane.
5. On the **Service Unit** tab, click **Create Group** in the lower part of the group list.



6. Enter a name in the field that appears and click **OK**. A service unit group is created.
7. Click the **Application** tab. On the **Application** tab, click **Create Group** in the lower part of the group list. Enter a name in the field that appears and click **OK**. An application group is created.

The following table describes the operations that you can perform on the created groups.

Operation	Description
Change the name of a group	Click the  icon in the Actions column to change the name of a group.
Delete a group	Click the  icon in the Actions column to delete a group.

9.16.5.2. Create a service unit

9.16.5.2.1. Create a metadata set

By using metadata, you can configure and manage the schema mapping from NoSQL databases to relational databases to ensure the consistency of data sources. This topic describes how to create metadata sets for HBase, Elasticsearch, and MongoDB data sources.

Context

Before you create a service unit based on an HBase, an Elasticsearch, or a MongoDB data source, you must create a metadata set for the corresponding data source.

Create a metadata set for an HBase data source

1. [Log on to the Dataphin console.](#)
2. On the Dataphin homepage, click **Services** in the top navigation bar.
3. On the Service page, click **Administration** in the top navigation bar.
4. On the Administration page, click **Metadata Management** in the left-side navigation pane.
5. On the **Metadata Management** page, click **Create Metadata** in the upper-right corner.
6. On the **Create Metadata Set** page, specify a source table and add rowkeys and columns as required.

Section	GUI element	Description
Data Table Configuration	Physical Data Table	The configuration that specifies the source table. Select HBASE, a data source, and a physical table from the drop-down lists in sequence.
Rowkey Configuration	Delimiters	The delimiter used to separate different rowkeys.
	Rowkey Field Name	The name of the rowkey.
	Data Type	The data type of the rowkey.
	Row Key Description	The description of the rowkey.
	Actions	The operations that you can perform on the rowkey. You can click the  icon in the Actions column to delete a field.
Column Field Configuration	Column Family Name	The name of the column family to which the column belongs.
	Column Qualifier Name	The name of the column.
	Data Type	The data type of the column.
	Description	The description of the column.
	Actions	The operations that you can perform on the column. You can click the  icon in the Actions column to delete a column.

7. Click **Submit**.

Create a metadata set for an Elasticsearch data source

1. On the **Metadata Management** page, click **Create Metadata** in the upper-right corner.

2. On the **Create Metadata Set** page, specify a source table and add fields as required.

Section	GUI element	Description
Data Table Configuration	Physical Data Table	The configuration that specifies the source table. Select ES, a data source, and a physical table from the drop-down lists in sequence.
Field Configuration	Field Name	The name of the field. To configure a nested field, separate multiple field names with periods (.).
	Data Type	The data type of the field.
	Description	The description of the field.
	Actions	The operations that you can perform on the field. You can click the  icon in the Actions column to delete a field.

3. Click **Submit**.

Create a metadata set for a MongoDB data source

The configurations for creating a metadata set for a MongoDB data source are similar to those for an Elasticsearch data source. For more information, see [Create a metadata set for an Elasticsearch data source](#).

9.16.5.2.2. Create a service unit from a physical table

Dataphin allows you to create a service unit to manage the metadata of a physical table. This topic describes how to create a service unit from a physical table.

Prerequisites

- A service unit group is created. For more information, see [Create a group](#).
- The corresponding metadata set is created if the source physical table resides in an HBase, an Elasticsearch, or a MongoDB data source. For more information, see [Create a metadata set](#).

Context

The fields of a service unit can be referenced by multiple API operations.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Service Units** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. In the left-side navigation pane, click **Service Units**.
3. On the **Service Units** page, click **Create Service Unit** in the upper-right corner. The Create Service Unit page appears.
4. Select a service unit type.
 - i. In the Service Unit Type Selection step, select **Single Physical Table Service Unit**.
 - ii. Click **Next**.
5. Configure the service unit.

i. In the **Service Unit Basic Information Configuration** step, set the parameters as required.

Parameter	Description
Service unit name	The name of the service unit.
Service Unit Group	The group to which the service unit belongs.
Physical Data Table	The source table for the service unit. To specify a source table, perform the following steps: <ol style="list-style-type: none"> a. Select a data source type. b. Select a data source. c. Select a table.
Service unit description	The description of the service unit.

ii. Click **Next**.

6. Configure the fields of the service unit.

- i. In the **Service Unit Field Configuration** step, configure the fields of the service unit as required.
- ii. Click **Data Preview** to preview the data of the service unit to create.
- iii. Click **Identify and Publish in the Staging Environment** to create and publish the service unit in the staging environment.

7. On the **Service Units** page, find the created service unit, move the pointer over the  icon in the **Actions** column, and then select **Publish** to publish the service unit to the production environment.

The following table describes the operations that you can perform on the created service units.

Operation	Description
Modify the configuration of a service unit	Click the  icon in the Actions column to modify the configuration of a service unit.
Delete a service unit	<p>Move the pointer over the More icon in the Actions column and select the  icon to delete a service unit.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p> Note</p> <ul style="list-style-type: none"> ○ You can delete only unpublished service units. ○ To delete a service unit, you must be the owner of the service unit or the super administrator. </div>
Unpublish a service unit	Move the pointer over the  icon in the Actions column and select Unpublish to unpublish a service unit that has been previously published to the production environment.

Operation	Description
Manage the members of a service unit	<p>Click the  icon in the Actions column. The Service Unit Member Management page of the service unit appears.</p> <ul style="list-style-type: none"> ◦ Click Add Member. In the Add Members to Service Unit dialog box, select the name of the account to be added as a member and the user type from the Select Member and Select Role drop-down lists, respectively. Then, click Submit to add a member to the service unit. ◦ Click the  icon in the Actions column and select a member from the Transfer Ownership To drop-down list to transfer the ownership of the service unit to the selected member. ◦ Click the  icon in the Actions column to delete a member from the service unit. ◦ To delete multiple members at a time, select the members and click Delete Members in the upper-right corner.

9.16.5.2.3. Create a service unit from multiple physical tables

Dataphin allows you to create a service unit to manage the metadata of multiple physical tables. This topic describes how to create a service unit from multiple homogeneous or heterogeneous physical tables.

Prerequisites

- A service unit group is created. For more information, see [Create a group](#).
- The corresponding metadata sets are created if the source physical tables reside in HBase, Elasticsearch, or MongoDB data sources. For more information, see [Create a metadata set](#).

Context

The fields of a service unit can be referenced by multiple API operations.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Service Units** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. In the left-side navigation pane, click **Service Units**.
3. On the **Service Units** page, click **Create Service Unit** in the upper-right corner. The Create Service Unit page appears.
4. Select a service unit type.
 - i. In the Service Unit Type Selection step, select **Multiple Physical Table Service Unit**.
 - ii. Click **Next**.
5. Configure the service unit.

i. In the **Service Unit Basic Information Configuration** step, set the parameters as required.

GUI element	Description
Service unit name	The name of the service unit.
Service Unit Group	The group to which the service unit belongs.
Service unit description	The description of the service unit.
Add Physical Table	<p>To add a physical table, perform the following steps:</p> <ol style="list-style-type: none"> In the Mounted Physical Table List section, click Add Physical Table. In the Add Physical Table dialog box, select a data source type, a data source, and a table in sequence to specify a source table. If the required data source is unavailable, click Create Data Source to go to the Planning page and create a data source. For more information, see Create a MaxCompute data source. In the No field is selected section, select the fields of the specified physical table as required and click the  icon to add the fields to the Selected Field section. To remove a field from the Selected Field section, select the field and click the  icon. Click OK.

By default, the system sets the first physical table that you add as the primary table. You can click the  icon in the **Primary Table** column to change the primary table. You can also perform the following operations on the added physical tables:

- Click the  icon in the **Actions** column to modify the configuration of a physical table.
- Click the  icon in the **Actions** column to delete a physical table.

ii. Click **Next**.

6. Configure the fields of the service unit.

i. In the **Service Unit Field Configuration** step, configure the fields of the service unit as required.

Section	Parameter	Description
Service Unit Association Field Configuration	Service Unit Field	The name of the field in the service unit.
	Physical table name. Associated Field	The name of the association field. An association field is used to associate the physical tables that you added.
	Service Unit Association Field Type	The data type of the association field.
	Actions	The operations that you can perform on the association field. You can click the  icon in the Actions column to delete a configured association field. Note You cannot delete the association field in the first line.
Service Unit Field Configuration	Service Unit Field Type	The data type of the field of the service unit. Dataphin automatically selects the data type of each field based on the data type of the field in the source table.
	Service Unit Field	The name of the field. Dataphin automatically generates the name for each field based on the name of the field in the source table.

ii. Click **Data Preview** to preview the data of the service unit to create.

iii. Click **Identify and Publish in the Staging Environment** to create and publish the service unit in the staging environment.

7. On the **Service Units** page, find the created service unit, move the pointer over the  icon in the **Actions** column, and then select **Publish** to publish the service unit to the production environment.

The following table describes the operations that you can perform on the created service units.

Operation	Description
Modify the configuration of a service unit	Click the  icon in the Actions column to modify the configuration of a service unit.

Operation	Description
Delete a service unit	<p>Move the pointer over the More icon in the Actions column and click the  icon to delete a service unit.</p> <div style="background-color: #e0f2f7; padding: 10px; border: 1px solid #ccc;"> <p> Note</p> <ul style="list-style-type: none"> ○ You can delete only unpublished service units. ○ To delete a service unit, you must be the owner of the service unit or the super administrator. </div>
Unpublish a service unit	<p>Move the pointer over the  icon in the Actions column and select Unpublish to unpublish a service unit that has been previously published to the production environment.</p>
Manage the members of a service unit	<p>Click the  icon in the Actions column. The Service Unit Member Management page of the service unit appears.</p> <ul style="list-style-type: none"> ○ Click Add Member. In the Add Members to Service Unit dialog box, select the name of the account to be added as a member and the user type from the Select Member and Select Role drop-down lists, respectively. Then, click Submit to add a member to the service unit. ○ Click the  icon in the Actions column and select a member from the Transfer Ownership To drop-down list to transfer the ownership of the service unit to the selected member. ○ Click the  icon in the Actions column to delete a member from the service unit. ○ To delete multiple members at a time, select the members and click Delete Members in the upper-right corner.

9.16.5.3. Create an API operation

9.16.5.3.1. Create an API operation in template wizard mode

Dataphin allows you to create an API operation by specifying a service unit and configuring the mapping between the fields of the service unit and the request and response parameters of the API operation on the GUI. This topic describes how to create an API operation in template wizard mode.

Prerequisites

- A service unit group is created. For more information, see [Create a group](#).
- A service unit is created. For more information, see [Create a service unit from multiple physical tables](#) and [Create a service unit from a physical table](#).

Procedure

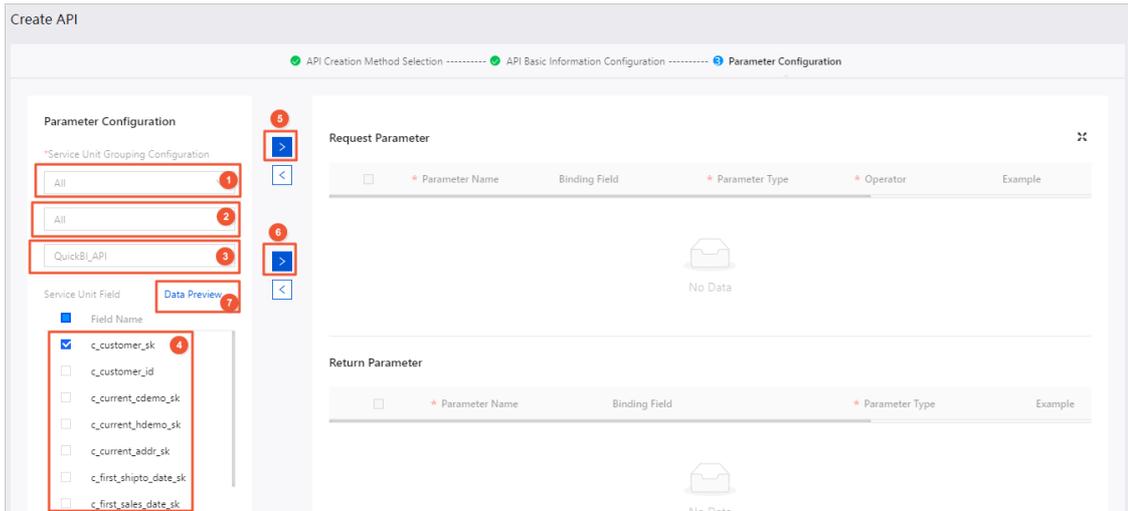
1. [Log on to the Dataphin console](#).

2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the APIs page, click **Create API** in the upper-right corner.
4. Select a creation method.
 - i. In the **API Creation Method Selection** step, select **Template Wizard Mode**.
 - ii. Click **Next**.
5. Configure the API operation.
 - i. In the **API Basic Information Configuration** step, set the parameters as required

Parameter	Description
API Name	The name of the API operation. The name can contain letters, digits, and underscores (_).
Request Method	The method for calling the API operation. Valid values: <ul style="list-style-type: none"> ▪ GET ▪ LIST
Data Update Frequency	The frequency at which data is synchronized by using the API operation. Valid values: <ul style="list-style-type: none"> ▪ Daily ▪ Hourly ▪ Minutely
Description	The description of the API operation.
Protocol	The protocol that is used by the API operation to synchronize data. Valid values: <ul style="list-style-type: none"> ▪ HTTP ▪ HTTPS

- ii. Click **Next Step**.
6. Configure the request and response parameters for the API operation.

- i. In the **Parameter Configuration** step, set the parameters as required.



No.	GUI element	Description
1	Service Unit Grouping Configuration	Click the <input type="checkbox"/> icon and select a service unit group.
2		Click the <input type="checkbox"/> icon and select a service unit type.
3		Click the <input type="checkbox"/> icon and select a service unit.
4	Service Unit Field	Select fields of the service unit.
5	N/A	After you complete the preceding configuration in sections marked as 1 to 4, click the <input type="checkbox"/> icon marked as 5 to add the selected fields to the Request Parameter section. Then, set Parameter Name , Parameter Type , Operator , Example , Description , and Required for the added request parameters.
6	N/A	After you complete the preceding configuration in sections marked as 1 to 4, click the <input type="checkbox"/> icon marked as 6 to add the selected fields to the Return Parameter section. Then, set Parameter Name , Parameter Type , Example , and Description for the added response parameters.
7	Data Preview	Click Data Preview to preview the data of the selected service unit.

- ii. Click **Identify and Publish in the Staging Environment** to create and publish the API operation in the staging environment.

9.16.5.3.2. Create an API operation in custom SQL mode

Dataphin allows you to create an API operation by specifying a service unit and writing an SQL statement. Dataphin parses the request and response parameters of the API operation based on the SQL statement. This topic describes how to create an API operation in custom SQL mode.

Prerequisites

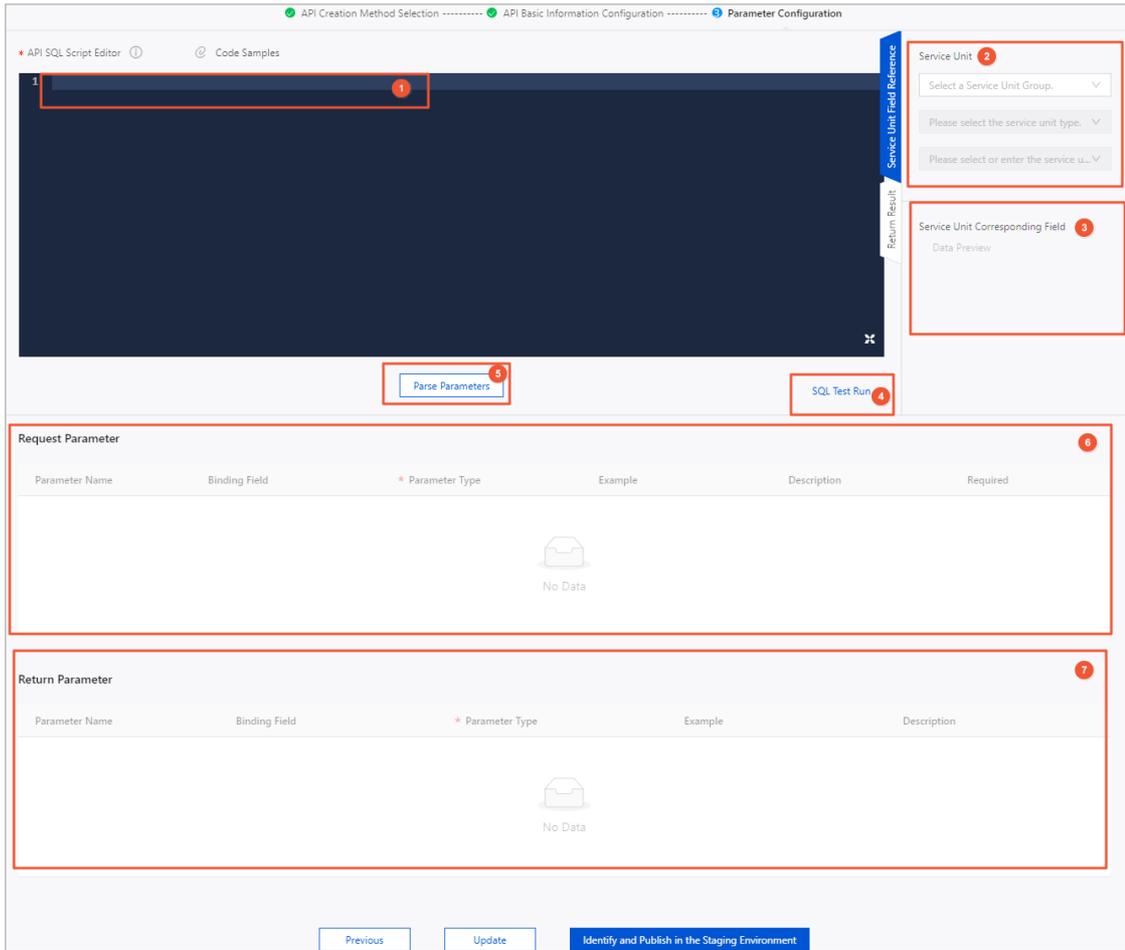
- A service unit group is created. For more information, see [Create a group](#).
- A service unit is created. For more information, see [Create a service unit from multiple physical tables](#) and [Create a service unit from a physical table](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the APIs page, click **Create API** in the upper-right corner.
4. Select a creation method.
 - i. In the **API Creation Method Selection** step, select **Custom SQL Mode**.
 - ii. Click **Next**.
5. Configure the API operation.
 - i. In the **API Basic Information Configuration** step, set the parameters as required

Parameter	Description
API Name	The name of the API operation. The name can contain letters, digits, and underscores (_).
Request Method	The method for calling the API operation. Valid values: <ul style="list-style-type: none"> ▪ GET ▪ LIST
Data Update Frequency	The frequency at which data is synchronized by using the API operation. Valid values: <ul style="list-style-type: none"> ▪ Daily ▪ Hourly ▪ Minutely
Description	The description of the API operation.
Protocol	The protocol that is used by the API operation to synchronize data. Valid values: <ul style="list-style-type: none"> ▪ HTTP ▪ HTTPS

- ii. Click **Next Step**.
6. Configure the request and response parameters for the API operation.
 - i. In the **Parameter Configuration** step, enter an SQL statement in the script editor and set the parameters as required.



No.	GUI element	Description
1	API SQL Script Editor	<p>Enter an SQL statement in the API SQL Script Editor section. For example, you can enter the following statement:</p> <pre>select user_id,pid,city_id,activity_time from ads1578301726 where user_id=\${user_id};</pre> <p>You can click the  icon to view the limits on the SQL statement.</p>
2	Service Unit	<p>To specify a service unit, perform the following steps:</p> <ol style="list-style-type: none"> Select a service unit group from the drop-down list. Select a service unit type from the drop-down list. Select a service unit from the drop-down list.
3	Data Preview	<p>Click Data Preview to preview the data of the selected service unit.</p>

No.	GUI element	Description
4	SQL Test Run	<p>To run the SQL statement, perform the following steps:</p> <ol style="list-style-type: none"> Click SQL Test Run. In the Specify Request Parameters dialog box, set Test Run Input to the value that you obtained when you previewed the data of the selected service unit. In this example, enter the value of the <code>user_id</code> field. Click Confirm. <ul style="list-style-type: none"> If SQL Status appears as Success, the SQL syntax, computing engine, and database engine passed the validation checks. If SQL Status appears as Error, the SQL syntax, computing engine, and database engine failed the validation checks.
5	Parse Parameters	Click Parse Parameters for the system to automatically generate request and response parameters based on the SQL statement that you entered in the API SQL Script Editor section.
6	Request parameters	<p>Specify the following fields for the request parameters:</p> <ul style="list-style-type: none"> Parameter Type Example Description Required
7	Return Parameter	<p>Specify the following fields for the response parameters:</p> <ul style="list-style-type: none"> Parameter Type Example Description

- Click **Identify and Publish in the Staging Environment** to create and publish the API operation in the staging environment.

9.16.5.4. Test an API operation

You can publish an API operation to the production environment only after it passes the test in the staging environment. This topic describes how to test an API operation.

Prerequisites

An API operation is created. For more information, see [Create an API operation in template wizard mode](#) and [Create an API operation in custom SQL mode](#).

Context

You can test API operations only in the staging environment. We recommend that you test API operations before you publish them to the production environment.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the **APIs** page, find the API operation that you want to test.
 - You can filter API operations by selecting a service unit or service unit group from the corresponding drop-down list.
 - Alternatively, you can enter a keyword in the search box to filter API operations whose names contain the keyword.
4. Move the pointer over the  icon in the Actions column and select **Test**.
5. On the **API Test** page, enter the input values for API-specific request parameters based on the example values in the **API Request Parameters** section. Enter the input values for common request parameters based on the example values in the **Public Request Parameter List** section. Select response parameters as required in the **Optional Response Parameters** section. Then, select a protocol from the **Protocol** drop-down list.
6. Click **Test**.
7. After the test is completed, view the test results of the API operation in the **Test Results** section. You can also click **View Error Code Table** in the lower part of the page to view error codes and their descriptions.

9.16.5.5. Publish an API operation

API operations can be called only after they are published to the production environment. The API Service page lists the API operations that have been published. This topic describes how to publish an API operation.

Prerequisites

An API operation is created. For more information, see [Create an API operation in template wizard mode](#) and [Create an API operation in custom SQL mode](#).

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the **APIs** page, find the API operation that you want to publish.
 - You can filter API operations by selecting a service unit or service unit group from the corresponding drop-down list.
 - Alternatively, you can enter a keyword in the search box to filter API operations whose names contain the keyword.
4. Move the pointer over the  icon in the Actions column and select **Publish**.

5. In the message that appears, click **OK** to publish the API operation to the production environment.

9.16.5.6. Unpublish an API operation

This topic describes how to unpublish an API operation.

Prerequisites

An API operation is published. For more information, see [Publish an API operation](#).

Context

If you find an API operation inappropriate, you can unpublish it. Unpublished API operations do not appear on the API Service page. After you unpublish an API operation, you can delete it from Development Center. For more information about how to delete an API operation, see [Delete an API operation](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the **APIs** page, find the API operation that you want to unpublish.
 - o You can filter API operations by selecting a service unit or service unit group from the corresponding drop-down list.
 - o Alternatively, you can enter a keyword in the search box to filter API operations whose names contain the keyword.
4. Move the pointer over the  icon in the Actions column and select **Unpublish**.

 **Note** You can only unpublish API operations that are not being used by any application.

5. In the message that appears, click **OK**.

9.16.5.7. Delete an API operation

This topic describes how to delete an API operation from Development Center.

Prerequisites

An API operation is published. For more information, see [Publish an API operation](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Development Center** in the left-side navigation pane.
3. On the **APIs** page, find the API operation that you want to delete.

- You can filter API operations by selecting a service unit or service unit group from the corresponding drop-down list.
 - Alternatively, you can enter a keyword in the search box to filter API operations whose names contain the keyword.
4. Move the pointer over the  icon in the Actions column and select **Delete**.

 **Note** You can only delete API operations that are not being used by any application.

5. In the message that appears, click **OK**.

9.16.6. Use and manage an API operation

9.16.6.1. Create an application

You must use applications to call API operations in the Services module. An application supports multiple API operations. This topic describes how to create an application.

Prerequisites

An application group is created. For more information, see [Create a group](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Applications** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **Applications** under **Application Center** in the left-side navigation pane.
3. On the **Applications** page, click **Create Application** in the upper-right corner.
4. In the **Create Application** dialog box, set the **Application Group** and **Application Name** parameters as required.
5. Click **Confirm**. The following table describes the operations that you can perform on the created applications.

Operation	Description
Modify the configuration of an application	Click the  icon in the Actions column to modify the configuration of an application.

Operation	Description
Manage the members of an application	<p>Click the  icon in the Actions column. The Application Member Management page of the application appears.</p> <ul style="list-style-type: none"> ○ Click Add Member. In the Add Member dialog box, select the name of the account to be added as a member and the user type from the Member Name and Select Role drop-down lists, respectively. Then, click Submit to add a member to the application. <ul style="list-style-type: none"> ■ The members of an application can apply for permissions on API operations for the application and call the API operations that the application is authorized to access. ■ The administrator of an application can apply for permissions on API operations for the application, call the API operations that the application is authorized to access, and remove API operations from the application. ○ Click the  icon in the Actions column and select a member from the Transfer Ownership To drop-down list to transfer the ownership of the application to the selected member. ○ Click the  icon in the Actions column to delete a member from the application. ○ To delete multiple members at a time, select the members and click Remove in the upper-right corner.
Go to the API Service page	Click the  icon in the Actions column. The API Service page appears, where you can view the published API operations.
Delete an application	Click the  icon in the Actions column to delete an application.
View the key and secret of an application	On the Applications page, view the key and secret of an application in the AppKey and AppSecret columns, respectively.

9.16.6.2. View an API operation

Before you apply for permissions on an API operation for an application, we recommend that you view the details of the API operation and determine whether it meets your business requirements. This topic describes how to view an API operation on the API Service page.

Prerequisites

An API operation is published. For more information, see [Publish an API operation](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **API Service** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **API Service** in the top navigation bar.
3. On the **API Service** page, find the API operation that you want to view and click **API Documentation** in the **Documentation** column.

4. On the **API Documentation** page, view the information in the **Basic Information**, **API Request Parameters**, **Public Request Parameter List**, and **Response Parameters** sections and determine whether the API operation meets your business requirements.

9.16.6.3. Apply for permissions on an API operation

Before you call an API operation by using an application, the application must be authorized to access the API operation. This topic describes how to apply for the permission to query fields of an API operation for an application.

Prerequisites

An application is created. For more information, see [Create an application](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **API Service** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **API Service** in the top navigation bar.
3. On the API Service page, find the API operation on which you want to apply for permissions and click **Request Now** in the **Request Status** column.
4. In the Select Application and API step of the **API Permission Request** wizard, set the **Application** and **API** parameters as required.
5. Click **Next**.
6. In the **Apply for API Permissions** step, set the parameters as required.

Section	Parameter	Description
Object Information	N/A	In this section, you can view the names of the application and API operation, and the groups to which they belong.
Authorized Fields	N/A	Select the fields on which you want to apply for permissions. You can enter a keyword in the search box to search for the fields whose names contain the keyword.
Permission Configuration	Validity Period	The validity period of the query permission. Valid values: <ul style="list-style-type: none">◦ 30 Days◦ 90 Days◦ 180 Days◦ Long-term◦ Custom Expiration Date Dataphin allows only application users of an application to apply for the permission to query fields of an API operation for the application.
	Reason for Application	The reason for applying for the permission to query fields of the API operation.

7. Click **OK**.

9.16.6.4. Debug an API operation

Before you call an API operation by using an application, you must debug the API operation to ensure data security and stability. This topic describes how to debug an API operation.

Prerequisites

- An application is granted the query permission on the API operation that you want to debug. For more information, see [Apply for permissions on an API operation](#).
- The key and secret of the application that is granted the query permission are obtained.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **APIs** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Workbench** in the top navigation bar.
 - iii. On the Workbench page, click **APIs** under **Application Center** in the left-side navigation pane.
3. Find the API operation that you want to debug and click the  icon in the **Actions** column. The **API Debugging** page appears.
4. On the **API Debugging** page, set the parameters as required.

Section	Parameter	Description
API Request Parameters	Input Value	The value of the required request parameter for debugging the API operation.
Public Request Parameter List	Input Value	The value of the common request parameter for debugging the API operation. <ul style="list-style-type: none"> ◦ Input Value of appkey: the key of the application for debugging the API operation. ◦ Input Value of appsecret: the secret of the application for debugging the API operation.
Response Parameters	Protocol	In the Response Parameters section, select the response parameters and the protocol to be used for the debugging as required.

5. Click **Debug**.
6. After the debugging is completed, view the result in the **Results** section. You can also click **View Error Code Table** in the lower part of the page to view error codes and their descriptions. If the debugging fails, you can go to the APIs page to remove the API operation from the application. To remove an API operation, find the API operation and click the  icon in the **Actions** column. In the message that appears, click **OK**. You can also apply for permissions on specific fields of an API operation on the APIs page. To apply for field permissions, find the API operation and click the  icon in the **Actions** column. On the **API Permission Request** page, set the parameters as required and click **OK**.

9.16.6.5. Call an API operation

This topic describes how to use an application to call an API operation that has been published to the production environment.

Prerequisites

- The network is configured. For more information, see [Configure the network](#).
- An application is granted the query permission on an API operation. For more information, see [Apply for permissions on an API operation](#).
- The key and secret of the application are obtained. For more information, see [Create an application](#).
- The API operation is debugged. For more information, see [Debug an API operation](#).

Context

You can call an API operation by using Services SDK for Java or Postman. We recommend that you use Services SDK for Java to call an API operation. You can use Postman only when you debug or develop an API operation.

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Call Examples** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the **Service** page, click **Administration** in the top navigation bar.
 - iii. On the **Administration** page, click **Call Examples** in the left-side navigation pane.
3. On the **API Call Examples** tab, view the sample code for using Services SDK for Java to call an API operation.
4. Click **Download SDK** in the upper-right corner to download the code package.
5. Modify the downloaded code based on the instructions on the **API Call Examples** tab. Then, you can call the API operation by using the specified application.

9.16.7. Monitor an API operation

Dataphin monitors API requests on a daily basis. You can view the statistics of the API operations that you have published on the Monitoring page. This topic describes how to view the request details of, set a request limit on, and configure alert rules for an API operation.

View the request details of an API operation

1. [Log on to the Dataphin console](#).
2. Go to the **Monitoring** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the **Service** page, click **Administration** in the top navigation bar.
 - iii. On the **Administration** page, click **Monitoring** in the left-side navigation pane.
3. On the **Monitoring** page, click **Details** in the **Actions** column of the API operation whose details you want to view. On the API Monitoring page, you can view the statistics of the API requests in the last hour.

Set a request limit on an API operation

 **Note** To set a request limit on an API operation, you must be the developer of the API operation or the super administrator.

1. On the **Monitoring** page, click **Limit** in the **Actions** column of the API operation on which you want to

set a request limit.

2. In the **Configure API Call Limit** dialog box, set the parameters as required.

GUI element	Description
Limit Status	Specifies whether to set a request limit on the API operation.
Time Unit	<p>The time granularity of the request limit to be set on the API operation. Valid values:</p> <ul style="list-style-type: none"> hour sec min day
API Call Limit	<p>The maximum number of API requests that can be handled at the specified granularity. Assume that your enterprise has purchased the default specification for the Services module, which supports handling 500 API requests per second. In this case, if you set Time Unit to sec, you must specify a value less than or equal to 500 for the API Call Limit parameter.</p>
Application Call Limit	<p>The maximum number of API requests sent by each involved application that can be handled at the specified granularity. Note the following rules when you set the Application Call Limit parameter:</p> <ul style="list-style-type: none"> The specified value of Application Call Limit applies to all the applications that are authorized to access the API operation. The value of Application Call Limit must be less than or equal to that of API Call Limit. The API Call Limit parameter has a higher priority than the Application Call Limit parameter. Assume that the request limit on an API operation is 1,000 requests per second and that on each of the three applications that are authorized to access the API operation is 500 requests per second. The sum of the permitted requests per second for the three applications exceeds the request limit on the API operation. However, with a higher priority, the request limit on the API operation determines the upper limit of requests. If you set Application Call Limit to a value greater than that of API Call Limit, an error message is returned.
Add Application	<p>Click Add Application and set the Application and Call Limit parameters to set a different request limit on a specific application. You can also click the  icon in the Actions column of an application to remove the request limit from the application. Note the following rules when you set the Call Limit parameter:</p> <ul style="list-style-type: none"> The value of Call Limit for a specific application must be less than or equal to that of API Call Limit. The Call Limit parameter of a specific application has a higher priority than the Application Call Limit parameter. Assume that the request limit on a specific application is 0 or 200 API requests per second and that on each of the other applications that are authorized to access the same API operation is 100 API requests per second. In this case, with a higher priority, the request limit on the specific application determines the upper limit of requests for itself. If you set the Call Limit parameter of a specific application to a value greater than that of API Call Limit, an error message is returned.

3. Click **OK**.

Configure alert rules for an API operation

1. On the **Monitoring** page, click **Alert** in the **Actions** column of the API operation for which you want to configure alert rules.
2. On the **Alert** page, click **Create Rule** in the upper-right corner.
3. Set the parameters as required.

Parameter	Description
Monitoring Metric	The metric to be monitored for the API operation. Valid values: <ul style="list-style-type: none"> ◦ Number of Calls: the total number of API requests per unit time. ◦ Average Response time: the average response time of API requests per unit time. The average response time equals the total time for processing the API requests divided by the total number of API requests. ◦ Failed Calls (Percentage): the ratio of the number of failed API requests to the total number of API requests per unit time. ◦ Calls During Offline (Percentage): the ratio of the number of API requests that failed due to connection interruptions to the total number of API requests per unit time.
Time Unit	The time interval at which monitoring data is queried. Valid values: 1min, 5min, 10min, 30min, and 60min.
Operator	The comparison operator that is used to compare the value of the monitored metric with the specified value of Trigger Threshold . Valid values: <=, <, >, >=, =, != .
Trigger Threshold	The threshold cross which an alert will be triggered.
Notification	The method that is used to send alert notifications to the recipient. Valid values: <ul style="list-style-type: none"> ◦ Email ◦ Cellphone ◦ SMS ◦ DingTalk
Recipient	The user to receive alert notifications.
Silent Period	The period of time in which an alert notification is not sent repeatedly. This helps avoid notification redundancy.

4. Click **OK**. On the **Alert** page, you can view the basic information about the selected API operation and the alert rules that have been created for the API operation. You can perform the following operations on existing alert rules:
 - Click the  icon in the **Actions** column of an alert rule to modify the rule.
 - Click the  icon in the **Actions** column of an alert rule to delete the rule.

9.16.8. Create a Dataphin data source

In the Services module, you can create Dataphin data sources from logical or physical tables for applications to access. This topic describes how to create a Dataphin data source.

Context

After you create Dataphin data sources from logical or physical tables, you can further develop these data resources based on the specific business scenarios. This allows you to enjoy a secure, stable, and easy-to-use data sharing service at low costs.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Dataphin Data Sources** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Administration** in the top navigation bar.
 - iii. On the Administration page, click **Dataphin Data Sources** in the left-side navigation pane.
3. On the **Dataphin Data Sources** page, click **Create Data Source** in the upper-right corner.
4. On the **Create Data Source** page, set the parameters as required.

Section	Parameter	Description
Basic Information	Data Source Name	The name of the data source to create.
	Data Source Description	The description of the data source.
Logical Table Selection	Unselected Logical Tables	Lists the logical tables that you are authorized to access. To specify the tables for creating the data source, select the tables and click the  icon to move the tables to the Selected Logical Tables section.
	Selected Logical Tables	Lists the logical tables that you specify for creating the data source. To remove tables from the list, select the tables and click the  icon to move the tables to the Unselected Logical Tables section.
Physical Table Selection	Unselected Physical Tables	Lists the physical tables that you are authorized to access. To specify the tables for creating the data source, select the tables and click the  icon to move the tables to the Selected Logical Tables section.
	Selected Physical Tables	Lists the physical tables that you specify for creating the data source. To remove tables from the list, select the tables and click the  icon to move the tables to the Unselected Physical Tables section.
Computing Space	Select Project for Data Source Computing	The Dataphin project for the computing work of the data source to be created.

5. Click **Create Data Source**. After the data source is created, it appears on the Dataphin Data Sources page. This page lists the data sources that applications can use. The following table describes the operations that you can perform on the created data sources.

Operation	Description
Modify the configuration of a data source	Click the  icon in the Actions column to modify the configuration of a data source.
Delete a data source	Click the  icon in the Actions column to delete a data source.

9.16.9. Use and manage a Dataphin data source

9.16.9.1. View a Dataphin data source

Before you use a Dataphin data source, we recommend that you view the details of the data source and determine whether it meets your business requirements. This topic describes how to view a Dataphin data source.

Prerequisites

A Dataphin data source is created. For more information, see [Create a Dataphin data source](#).

Procedure

1. [Log on to the Dataphin console](#).
2. Go to the **Dataphin Data Sources** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Administration** in the top navigation bar.
 - iii. On the Administration page, click **Dataphin Data Sources** in the left-side navigation pane.
3. On the Dataphin Data Sources page, find the data source that you want to view and click the  icon in the **Actions** column. The **View Data Source** page appears.
4. On the **View Data Source** page, view the information in the **Basic Information**, **Logical Table Information**, **Physical Table Information**, and **Computing Space Information** sections and determine whether the data source meets your business requirements.

9.16.9.2. Use a Dataphin data source

Dataphin provides the JDBC interface for you to use Dataphin data sources. You must download the JAR package of the Dataphin JDBC driver and import the JAR package by using a JDBC client or by writing Java code. Then, you can use Dataphin data sources. This topic describes how to use a Dataphin data source.

Prerequisites

- A Dataphin data source that meets your business requirements is found. For more information, see [View a Dataphin data source](#).
- The network is configured. For more information, see [Configure the network](#).
- The permission to query the Dataphin data source is granted by the super administrator.
- The JDBC URL of the Dataphin data source is granted by the super administrator.
- The **AccessKey** of the Dataphin data source is obtained.

Procedure

1. [Log on to the Dataphin console.](#)
2. Go to the **Call Examples** page.
 - i. On the Dataphin homepage, click **Services** in the top navigation bar.
 - ii. On the Service page, click **Administration** in the top navigation bar.
 - iii. On the Administration page, click **Call Examples** in the left-side navigation pane.
3. On the **Call Examples** page, click the **Dataphin Data Source Usage Examples** tab.
4. On the Dataphin Data Source Usage Examples tab, click **Download JDBC JAR Package** in the upper-right corner to download the Dataphin JDBC driver.
5. Use a JDBC client or write Java code based on the instructions on the **Dataphin Data Source Usage Examples** tab to use the data source.

10. Elasticsearch (on ECS)

10.1. What is Apsara Stack Elasticsearch?

Elasticsearch is a Lucene-based data search service. It provides a distributed, multi-tenancy, full-text search engine that uses a RESTful web interface. Elasticsearch is an open source product that is developed in Java and released under the Apache License. It is a popular search engine for enterprises. Elasticsearch is designed to serve cloud computing for real-time searches. It is stable, reliable, fast, and easy to install and use.

Apsara Stack Elasticsearch is released in 5.5.3 and 6.3.2 versions. Apsara Stack Elasticsearch is ideal for data analytics and searches. Based on open source Elasticsearch, Apsara Stack Elasticsearch provides enterprise-grade access control. Each Apsara Stack Elasticsearch cluster is composed of multiple nodes. The management nodes of Apsara Stack Elasticsearch support high availability. Management node failures do not affect the running of Apsara Stack Elasticsearch. In addition, Apsara Stack Elasticsearch can store audit logs and automatically dump them to the specified directory on a specific node for long-term storage and management.

Built-in plug-ins provided by Apsara Stack Elasticsearch include but are not limited to:

- **IK analyzer:** an open source, lightweight Chinese tokenizer kit developed in Java. It is a popular plug-in for Chinese tokenization in open source communities.
- **Smart Chinese analysis plug-in:** the default Lucene Chinese tokenizer. It enables an Apsara Stack Elasticsearch cluster to be automatically scaled to hundreds of nodes and process petabytes of structured or unstructured data.
- **ICU analysis plug-in:** a Lucene ICU tokenizer. ICU is a set of stable, tested, powerful, easy-to-use libraries and provides Unicode support for applications on different platforms.
- **Japanese (Kuromoji) analysis plug-in:** a Japanese tokenizer.
- **Stempel (Polish) analysis plug-in:** a French tokenizer.
- **Mapper attachments type plug-in:** an attachment plug-in that can use Apache Tika to parse content in different types of files into strings.

10.2. Quick start

This topic shows you how to create an Elasticsearch instance based on ECS. It covers creating a VPC, creating a security group, creating an ECS instance, creating an Elasticsearch instance, and connecting to an Elasticsearch instance.

Create an Elasticsearch instance based on ECS

After you create an Elasticsearch instance, a VPC, and an ECS instance (in the same region as the Elasticsearch instance), the ECS instance can be used as the client. Then you can deploy a user program or run the curl command.

 **Note** If your Elasticsearch instance and ECS instance share the same VPC and region but reside in different zones, you must create a VSwitch in the zone where the ECS instance resides to ensure that the ECS instance can connect to your Elasticsearch instance.

10.2.1. Log on to the Elasticsearch console

This topic describes how to log on to the Elasticsearch console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, choose **Products > Big Data > Elasticsearch**.
5. Specify **Organization** and **Region** and click **Elasticsearch**.

10.2.2. Create an Elasticsearch cluster

This topic describes how to create an Elasticsearch cluster.

Prerequisites

- A Virtual Private Cloud (VPC) and a VSwitch are created.
For more information, see the "Quick start" topic in *Apsara Stack Virtual Private Cloud User Guide*.
- Elastic Compute Service (ECS) instances are sufficient.

Precautions

Before you create an Elasticsearch cluster, you must understand the main data sources of the Elasticsearch cluster. This helps you better plan the size and allocation of disk space.

If you specify a small disk size, Elasticsearch cluster logs occupy a relatively large proportion of disk space. An Elasticsearch cluster mainly stores the following types of data:

- User data that has been pushed to the Elasticsearch cluster.
- Index replicas. You can specify the number of replicas for each index but must make sure that each index has a minimum of one replica.

Procedure

1. Log on to the [Elasticsearch console](#).
2. On the page that appears, click **Create**.
3. Specify the required parameters.

Type	Parameter	Description
Region	Region	The region where the cluster is deployed.
	Zone	An Elasticsearch cluster can be deployed across zones. You may find multiple available zones from the Zone drop-down list. Select one or more appropriate zones as needed.
Cluster	Version	Valid values: 5.5.3 and 6.3.2.
	Network Type	The value of this parameter can only be VPC.
	VPC	Select the same VPC as your ECS instances.
	VSwitch	Select the same VSwitch as your ECS instances.
	Specification Family	Valid values: Local SATA, Local SSD, and Cloud Disk.
	Instance Specification	<ul style="list-style-type: none"> ◦ Local SATA 16-Core 64 GB and 8-Core 32 GB ◦ Local SSD 16-Core 64 GB and 8-Core 32 GB ◦ Cloud Disk 16-Core 64 GB, 8-Core 32 GB, and 4-Core 16 GB
	Count	The number of data nodes. You must purchase a minimum of two data nodes. A cluster that contains only two data nodes has a high risk of split-brain. Therefore, exercise caution when you set this parameter.
	Dedicated Master Node	To improve the stability of your cluster, we recommend that you purchase dedicated master nodes. Dedicated master nodes support the following specifications: 4-Core 16 GB, 8-Core 32 GB, and 16-Core 64 GB. Standard SSDs are supported for storage.
Client Node	For CPU-intensive services, we recommend that you purchase client nodes to share the CPU overheads of data nodes. This further improves the computing performance and service stability of your cluster. For example, if a large number of aggregation operations are performed, you can use client nodes to share the overheads. Client nodes support the following specifications: 4-Core 16 GB, 8-Core 32 GB, and 16-Core 64 GB. Ultra disks are supported for storage.	

Type	Parameter	Description
	Warm Node	<p>If your business includes both of the following index types, we recommend that you purchase warm nodes to implement the hot-warm architecture. This architecture improves the computing performance and service stability of your cluster.</p> <ul style="list-style-type: none"> ◦ Frequently queried or written indexes ◦ Infrequently queried or written indexes, typically indexes of records <p>Warm nodes support the following specifications: 4-Core 16 GB, 8-Core 32 GB, and 16-Core 64 GB. Ultra disks are supported for storage.</p> <p>You can select Warm Node on the buy page or configuration upgrade page to purchase warm nodes. After you purchase nodes, the system adds <code>-Enode.attr.box_type</code> to their startup parameters.</p> <ul style="list-style-type: none"> ◦ Data nodes: <code>-Enode.attr.box_type=hot</code> ◦ Warm nodes: <code>-Enode.attr.box_type=warm</code>
Storage	Disk Type	<p>Standard SSDs and ultra disks are supported.</p> <ul style="list-style-type: none"> ◦ A standard SSD provides a maximum of 2,048 GiB of storage space. Standard SSDs are ideal for online data analytics and searches that require high IOPS and fast responses. ◦ An ultra disk provides a maximum of 5,120 GiB of storage space. Ultra disks are cost-effective and are ideal for scenarios such as logging and analyzing large amounts of data. Ultra disks with the storage space greater than 2,560 GiB cannot be resized because these disks are designed to run in disk arrays or RAID 0.
	Storage Space per Data Node	<p>The storage space of each data node. It depends on the disk type. Unit: GiB.</p> <p>If the disk type is SSD Cloud Disk, the maximum value of this parameter is 2048.</p> <p>If the disk type is Ultra Disk, the maximum value of this parameter is 5120. If the volume of the data that you want to store exceeds 2,048 GiB, you can set this parameter to 2560, 3072, 3584, 4096, 4608, or 5120. If the storage space of the disk for a purchased cluster is less than 2,048 GiB, you can resize the disk to a maximum of 2,048 GiB.</p>
Password	Username	The value of this parameter can only be elastic.
	Password	The password of the Elasticsearch cluster. This password is also used to log on to the Kibana console.

Note
This parameter can be configured only when the Specification Family parameter is set to Cloud Disk.

4. Click **Create**.

You can then find the cluster in the cluster list. When **Active** is displayed for the cluster, the cluster is created.

10.2.3. Access an Elasticsearch cluster

This topic describes how to access an Elasticsearch cluster. You can access an Elasticsearch cluster from an Elastic Compute Service (ECS) instance or the Kibana console.

Prerequisites

You have completed the following operations:

- Create an Elasticsearch cluster.
- Create an ECS instance that resides in the same Virtual Private Cloud (VPC), region, and zone as the Elasticsearch cluster.
- Obtain the internal endpoint and Kibana console URL of the Elasticsearch cluster.

You can [log on to the Elasticsearch console](#), find the target cluster on the Instances page, and click its ID in the Instance ID/Name column. Then, obtain the preceding information from the **Basic Information** page.

Use an ECS instance to access an Elasticsearch cluster

1. Log on to the ECS instance over SSH and install the cURL tool.

Note For information about other methods that are used to log on to an ECS instance, see the "Quick start" topic in Elastic Compute Service (ECS) of Alibaba Cloud Apsara Stack Enterprise V3.12.0 User Guide - Cloud Essentials and Security.

2. Run the following command to connect to the internal endpoint of the Elasticsearch cluster:

```
curl http://<HOST>:<PORT>
```

- Set `<HOST>` to the internal endpoint of the Elasticsearch cluster. You can obtain the internal endpoint from the Basic Information page of the cluster.
- Set `<PORT>` to the port number of the Elasticsearch cluster. You can obtain the port number from the Basic Information page of the cluster. The default port is 9200.

If the connection is established, the result shown in the following figure is returned.

```
[root@iZ7t... ~]# curl http://es-cn-d...elasticsearch.aliyun-inc.com:9200
{
  "name" : "k7d3r4E",
  "cluster_name" : "es-cn-d...",
  "cluster_uuid" : "Fmo...",
  "version" : {
    "number" : "5.5.3",
    "build_hash" : "9305a5e",
    "build_date" : "2017-09-07T15:56:59.599Z",
    "build_snapshot" : false,
    "lucene_version" : "6.6.0"
  },
  "tagline" : "You Know, for Search"
}
```

Use the Kibana console to access an Elasticsearch cluster

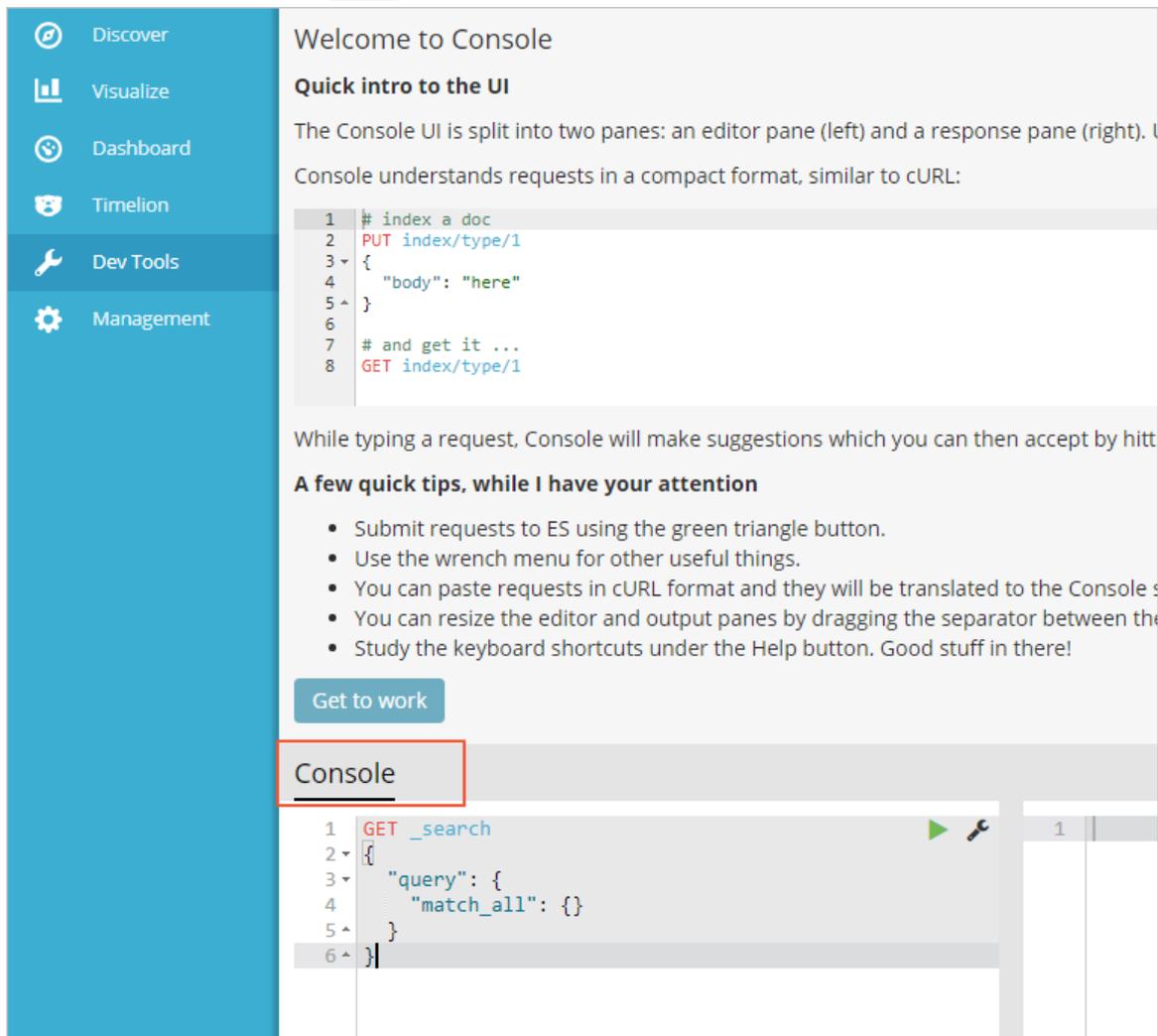
1. Enter the URL of the Kibana console in the address bar of your browser.

Notice The server where your browser is located must reside in the same VPC as the Elasticsearch cluster.

2. On the page that appears, enter the username and password and click Log in.

The username is elastic. The password is the one that is specified when you create the Elasticsearch cluster.

3. In the left-side navigation pane, click **Dev Tools**.
4. On the **Console** tab, run the `GET /` command to access the Elasticsearch cluster.



10.3. Manage clusters

This topic describes how to manage Elasticsearch clusters. Elasticsearch provides management features, such as **Kibana console**, **cluster restart**, and **refresh**.

10.3.1. Log on to the Kibana console

This topic describes how to log on to the Kibana console of an Elasticsearch cluster. Elasticsearch provides the Kibana console for you to scale your businesses. The Kibana console is seamlessly integrated into Elasticsearch. It allows you to view the status of your Elasticsearch cluster in real time and manage the cluster.

1. Log on to the [Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. On the **Basic Information** page, obtain the **Kibana console URL**. Then, enter the URL in the address bar of your browser.

 **Notice** The server where your browser is located must reside in the same Virtual Private Cloud (VPC) as the Elasticsearch cluster.

4. On the page that appears, enter the username and password and click Log in.

The username is elastic. The password is the one that is specified when you create the Elasticsearch cluster.

10.3.2. Restart an Elasticsearch cluster

This topic describes how to restart an Elasticsearch cluster. Two restart methods are supported: Restart and Force Restart. You can select a method based on your business scenario.

Precautions

- If the disk usage of an Elasticsearch cluster exceeds 85%, the status of the cluster may become abnormal (indicated by the color yellow or red). In this case, you must perform a forced restart.
- If the cluster is abnormal, we recommend that you do not perform the following operations on the cluster: node addition, node capacity expansion, disk resizing, restart, password reset, and configuration update. Perform these operations only after the status of the cluster becomes normal (indicated by the color green).
- If you update the configuration of an abnormal cluster that contains two or more nodes and the cluster remains in the Initializing state, submit a ticket.
- If a cluster contains only one node, services may become unavailable when you update the cluster configuration, restart or scale out/in the cluster, or reset the password of the cluster. In this case, create another Elasticsearch cluster and migrate your services to the new cluster.

Procedure

1. Log on to the [Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. On the **Basic Information** page, click **Restart Instance**.
4. In the **Restart Instance** dialog box, select a restart method.
 - **Restart**: This restart method does not affect the services of your cluster but is time-consuming. Before you select this method, make sure that each index of the cluster has a minimum of one replica.
 - **Force Restart**: This restart method may cause unstable services on your cluster but saves time.
5. Click **OK**.

 **Notice** Before the restart, make sure that the cluster is in a normal state. During the restart, the CPU utilization and memory usage of nodes in the cluster experience a surge. This may affect the stability of your services for a short period of time.

10.3.3. Refresh the information of an Elasticsearch cluster

This topic describes how to refresh the information of an Elasticsearch cluster. If part of the information in the console, such as the status of a newly created Elasticsearch cluster, is not refreshed in time, the console may fail to display the information. In this case, you can manually refresh the page to obtain the latest status.

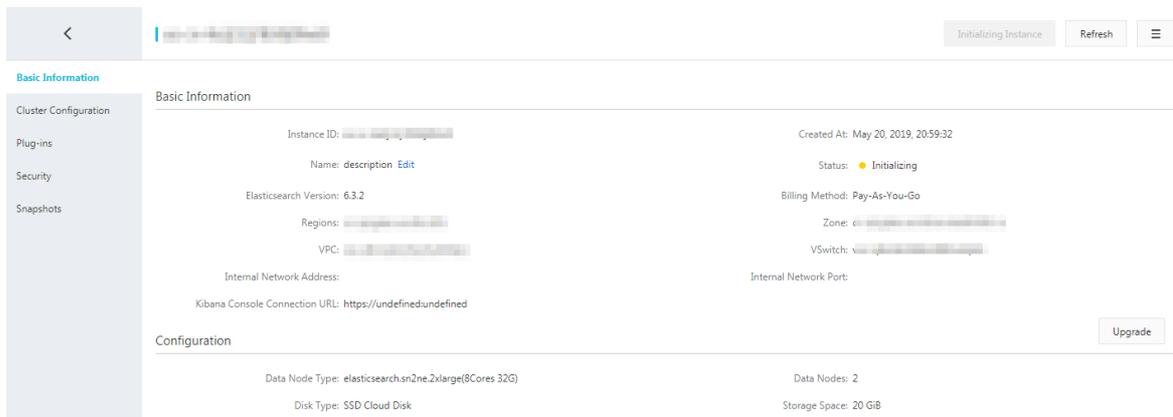
1. Log on to the [Elasticsearch console](#).

2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the upper-right corner of the **Basic Information** page, click **Refresh**.

10.3.4. View the basic information of an Elasticsearch cluster

This topic describes how to view the basic information of an Elasticsearch cluster.

1. Log on to the [Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. On the **Basic Information** page, view the basic information of the cluster.



Parameter	Description
Instance ID	The unique ID of the cluster.
Created At	The time when the cluster was created.
Name	The name of the cluster. By default, the name of a cluster is the same as its ID. Cluster names are configurable. You can search for clusters by name.
Status	The status of the cluster. A cluster has the following states: Active (green), Initializing (yellow), Unhealthy (red), Paused (red), and Expired (gray).
Elasticsearch Version	The version of the cluster. Valid values: 5.5.3 and 6.3.2.
Region	The region where the cluster resides.
Zone	The zone where the cluster resides.
VPC	The Virtual Private Cloud (VPC) where the cluster resides.
VSwitch	The VSwitch to which the cluster belongs.
Internal Network Address	The internal endpoint of the cluster. You can use an Elastic Compute Service (ECS) instance to connect to the internal endpoint of the cluster. Make sure that the ECS instance resides in the same VPC as the cluster.

Parameter	Description
Internal Network Port	The port used to connect to the cluster over an internal network. Elasticsearch supports the following ports: <ul style="list-style-type: none">Port 9200 for HTTPPort 9300 for TCP
Kibana Console Connection URL	The URL of the Kibana console of the cluster. You can use an ECS instance to connect to the URL. Make sure that the ECS instance resides in the same VPC as the cluster.

 **Note** You can also view the cluster configuration information on the Basic Information page. The information includes the data node specifications, number of data nodes, disk type, and storage space. You can determine whether to upgrade the cluster configuration based on the information and your business plan.

10.3.5. Upgrade the configuration of an Elasticsearch cluster

Apsara Stack Elasticsearch allows you to upgrade the specifications, storage space per node, and number of nodes for an Elasticsearch cluster. You cannot downgrade the configuration of an Elasticsearch cluster. This topic describes how to upgrade the configuration of an Elasticsearch cluster.

Procedure

1. Log on to the [Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the lower-right corner of the **Basic Information** page, click **Upgrade**.
4. On the **Update Configuration** page, modify the cluster configuration.

For information about parameters on this page, see [Create an Elasticsearch cluster](#).

Notice

- For each upgrade, you can modify only one of the following items: specifications, storage space per node, and the number of nodes.
- If your business requires a cluster configuration upgrade, evaluate your Elasticsearch cluster before the upgrade.

5. Click **Confirm Change**.

10.3.6. Configure an Elasticsearch cluster

10.3.6.1. Configure synonyms

You can configure synonyms for the synonym dictionary of an Apsara Stack Elasticsearch cluster. After you configure a tokenizer, new indexes in the cluster are tokenized based on the latest synonym dictionary. This topic describes how to configure synonyms.

Description

 Notice

- After you upload a synonym dictionary file to an Elasticsearch cluster, the system does not restart the cluster. The system sends the synonym dictionary file to all nodes in the cluster. The time that is required for the file to take effect depends on the number of nodes in the cluster.
- For example, the index-aliyun index is created based on the aliyun.txt synonym dictionary file. If you have uploaded a new synonym dictionary file to overwrite the existing aliyun.txt file, the index-aliyun index cannot automatically load the new dictionary file. If you want the index to automatically load the new dictionary file, we recommend that you disable the index before the upload and then enable the index after the upload. If an index that is created before the uploaded synonym dictionary file takes effect needs to use synonyms, you must reindex the data in the index and configure synonyms.

You can use a filter to configure synonyms. Sample code:

```
PUT /test_index
{
  "settings": {
    "index": {
      "analysis": {
        "analyzer": {
          "synonym": {
            "tokenizer": "whitespace",
            "filter": ["synonym"]
          }
        },
        "filter": {
          "synonym": {
            "type": "synonym",
            "synonyms_path": "analysis/synonym.txt",
            "tokenizer": "whitespace"
          }
        }
      }
    }
  }
}
```

- **filter** : specifies a **synonym** filter that contains the **analysis/synonym.txt** path. This path indicates the location of config.
- **tokenizer** : specifies a tokenizer that tokenizes synonyms. It is set to **whitespace** by default. Additional settings:
 - **ignore_case** : The default value is false.
 - **expand** : The default value is true.

Two synonym formats are supported: Solr and WordNet.

- Solr synonyms

Configuration example:

```
# Blank lines and lines starting with pound are comments.
# Explicit mappings match any token sequence on the LHS of "=>"
# and replace with all alternatives on the RHS. These types of mappings
# ignore the expand parameter in the schema.
# Examples:
i-pod, i pod => ipod,
sea biscuit, sea biscit => seabiscuit
# Equivalent synonyms may be separated with commas and give
# no explicit mapping. In this case the mapping behavior will
# be taken from the expand parameter in the schema. This allows
# the same synonym file to be used in different synonym handling strategies.
# Examples:
ipod, i-pod, i pod
foozball , foosball
universe , cosmos
lol, laughing out loud
# If expand==true, "ipod, i-pod, i pod" is equivalent
# to the explicit mapping:
ipod, i-pod, i pod => ipod, i-pod, i pod
# If expand==false, "ipod, i-pod, i pod" is equivalent
# to the explicit mapping:
ipod, i-pod, i pod => ipod
# Multiple synonym mapping entries are merged.
foo => foo bar
foo => baz
# is equivalent to
foo => foo bar, baz
```

You can also define synonyms in the filter, but you must use `synonyms` rather than `synonyms_path` .
Example:

```
PUT /test_index
{
  "settings": {
    "index": {
      "analysis": {
        "filter": {
          "synonym": {
            "type": "synonym",
            "synonyms": [
              "i-pod, i pod => ipod",
              "begin, start"
            ]
          }
        }
      }
    }
  }
}
```

We recommend that you use `synonyms_path` to define large synonym sets in the filter. Using `synonyms` to define large synonym sets increases the size of your cluster.

- WordNet synonyms

Configuration example:

```
PUT /test_index
{
  "settings": {
    "index": {
      "analysis": {
        "filter": {
          "synonym": {
            "type": "synonym",
            "format": "wordnet",
            "synonyms": [
              "s(100000001,1,'abstain',v,1,0).",
              "s(100000001,2,'refrain',v,1,0).",
              "s(100000001,3,'desist',v,1,0)."
            ]
          }
        }
      }
    }
  }
}
```

You can also use `synonyms_path` to define WordNet synonyms.

Use a synonym dictionary file to configure synonyms

1. Log on to the [Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane of the page that appears, click **Cluster Configuration**.
4. Upload a synonym dictionary file.
 - i. Click **Synonym Dictionary Configuration**.
 - ii. In the **Synonym Dictionary Configuration** pane, select the mode to upload a synonym dictionary file and click **Upload**. Then, select the .txt file that is generated based on the preceding rules.
 - iii. Click **Save**.

After the state of the Elasticsearch cluster becomes Active, you can use the synonym dictionary file. In this example, the `aliyun_synonyms.txt` file is uploaded. The file contains `begin, start`.

5. Configure and test the synonym dictionary.
 - i. Log on to the Kibana console.
For more information, see [Log on to the Kibana console](#).
 - ii. In the left-side navigation pane, click **Dev Tools**.

- iii. On the **Console** tab, run the following command to create an index:

```
PUT aliyun-index-test
{
  "index": {
    "analysis": {
      "analyzer": {
        "by_smart": {
          "type": "custom",
          "tokenizer": "ik_smart",
          "filter": ["by_tfr", "by_sfr"],
          "char_filter": ["by_cfr"]
        },
        "by_max_word": {
          "type": "custom",
          "tokenizer": "ik_max_word",
          "filter": ["by_tfr", "by_sfr"],
          "char_filter": ["by_cfr"]
        }
      },
      "filter": {
        "by_tfr": {
          "type": "stop",
          "stopwords": [" "]
        },
        "by_sfr": {
          "type": "synonym",
          "synonyms_path": "analysis/aliyun_synonyms.txt"
        }
      },
      "char_filter": {
        "by_cfr": {
          "type": "mapping",
          "mappings": ["| => |"]
        }
      }
    }
  }
}
```

- iv. Run the following command to configure the title synonym field:

```
PUT aliyun-index-test/_mapping/doc
{
  "properties": {
    "title": {
      "type": "text",
      "index": "analyzed",
      "analyzer": "by_max_word",
      "search_analyzer": "by_smart"
    }
  }
}
```

- v. Run the following command to verify synonyms:

```
GET aliyun-index-test/_analyze
{
  "analyzer": "by_smart",
  "text": "begin"
}
```

If the command is successfully executed, the following result is returned:

```
{
  "tokens": [
    {
      "token": "begin",
      "start_offset": 0,
      "end_offset": 5,
      "type": "ENGLISH",
      "position": 0
    },
    {
      "token": "start",
      "start_offset": 0,
      "end_offset": 5,
      "type": "SYNONYM",
      "position": 0
    }
  ]
}
```

- vi. Run the following commands to add data for further testing:

```
PUT aliyun-index-test/doc/1
{
  "title": "Shall I begin?"
}
```

```
PUT aliyun-index-test/doc/2
{
  "title": "I start work at nine."
}
```

- vii. Run the following command to perform a query test:

```
GET aliyun-index-test/_search
{
  "query": { "match": { "title": "begin" } },
  "highlight": {
    "pre_tags": ["<red>", "<bule>"],
    "post_tags": ["</red>", "</bule>"],
    "fields": {
      "title": {}
    }
  }
}
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 11,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 0.41048482,
    "hits": [
      {
        "_index": "aliyun-index-test",
        "_type": "doc",
        "_id": "2",
        "_score": 0.41048482,
        "_source": {
          "title": "I start work at nine."
        },
        "highlight": {
          "title": [
            "I <red>start</red> work at nine."
          ]
        }
      },
      {
        "_index": "aliyun-index-test",
        "_type": "doc",
        "_id": "1",
        "_score": 0.39556286,
        "_source": {
          "title": "Shall I begin?"
        },
        "highlight": {
          "title": [
            "Shall I <red>begin</red>?"
          ]
        }
      }
    ]
  }
}
```

Reference synonyms and use the IK dictionary for word splitting

1. Configure the `my_synonym_filter` synonym filter and a synonym dictionary.
2. Configure the `my_synonyms` analyzer and use the `ik_smart` IK analyzer to split words.
The `ik_smart` IK analyzer splits words and converts all letters into lowercase.

```
PUT /my_index
{
  "settings": {
    "analysis": {
      "analyzer": {
        "my_synonyms": {
          "filter": [
            "lowercase",
            "my_synonym_filter"
          ],
          "tokenizer": "ik_smart"
        }
      },
      "filter": {
        "my_synonym_filter": {
          "synonyms": [
            "begin,start"
          ],
          "type": "synonym"
        }
      }
    }
  }
}
```

3. Run the following command to configure the title synonym field:

```
PUT /my_index/_mapping/doc
{
  "properties": {
    "title": {
      "type": "text",
      "index": "analyzed",
      "analyzer": "my_synonyms"
    }
  }
}
```

4. Run the following command to verify synonyms:

```
GET /my_index/_analyze
{
  "analyzer": "my_synonyms",
  "text": "Shall I begin?"
}
```

If the command is successfully executed, the following result is returned:

```
{
  "tokens": [
    {
      "token": "shall",
      "start_offset": 0,
      "end_offset": 5,
      "type": "ENGLISH",
      "position": 0
    },
    {
      "token": "I",
      "start_offset": 6,
      "end_offset": 7,
      "type": "ENGLISH",
      "position": 1
    },
    {
      "token": "begin",
      "start_offset": 8,
      "end_offset": 13,
      "type": "ENGLISH",
      "position": 2
    },
    {
      "token": "start",
      "start_offset": 8,
      "end_offset": 13,
      "type": "SYNONYM",
      "position": 2
    }
  ]
}
```

5. Run the following commands to add data for further testing:

```
PUT /my_index/doc/1
{
  "title": "Shall I begin?"
}
```

```
PUT /my_index/doc/2
{
  "title": "I start work at nine."
}
```

6. Run the following command to perform a query test:

```
GET /my_index/_search
{
  "query": { "match": { "title": "begin" }},
  "highlight": {
    "pre_tags": ["<red>", "<bule>"],
    "post_tags": ["</red>", "</bule>"],
    "fields": {
      "title": {}
    }
  }
}
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 11,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 0.41913947,
    "hits": [
      {
        "_index": "my_index",
        "_type": "doc",
        "_id": "2",
        "_score": 0.41913947,
        "_source": {
          "title": "I start work at nine."
        },
        "highlight": {
          "title": [
            "I <red>start</red> work at nine."
          ]
        }
      },
      {
        "_index": "my_index",
        "_type": "doc",
        "_id": "1",
        "_score": 0.39556286,
        "_source": {
          "title": "Shall I begin?"
        },
        "highlight": {
          "title": [
            "Shall I <red>begin</red>?"
          ]
        }
      }
    ]
  }
}
```

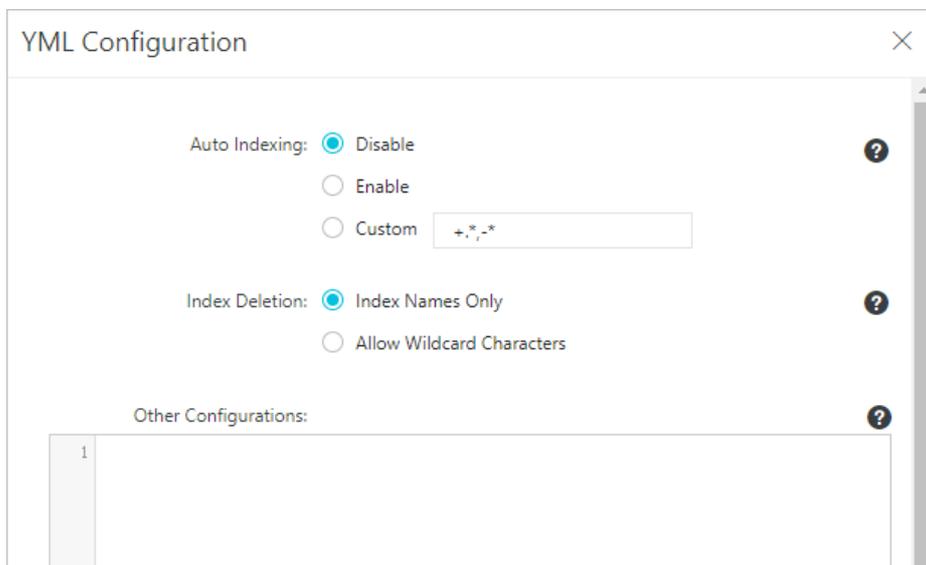
10.3.6.2. Perform configurations on YAML files

10.3.6.2.1. Configure a YAML file

The YAML file contains the configuration of an Elasticsearch cluster. You can use the file to modify the configuration of the cluster. This topic describes how to configure a YAML file.

Procedure

1. [Log on to the Elasticsearch console.](#)
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane of the page that appears, click **Cluster Configuration**.
4. On the page that appears, click **Modify Configuration** on the right side of **YAML Configuration**.
5. In the **YAML Configuration** pane, specify the required parameters.



Parameter	Description
Auto Indexing	<p>This parameter specifies whether to automatically create an index when a new document is uploaded to an Elasticsearch cluster but no index has been created. Valid values:</p> <ul style="list-style-type: none"> ◦ Disable: The system does not automatically create an index. This value is the default value. ◦ Enable: The system automatically creates an index. <div style="background-color: #e1f5fe; padding: 5px; border: 1px solid #cfcfcf;"> <p>Notice We recommend that you disable Auto Indexing because indexes created by this feature may not meet your business requirements.</p> </div> <ul style="list-style-type: none"> ◦ Custom: The system automatically creates a custom index. For example, if you specify <code>-an*,+a,-*</code>, the system automatically creates only indexes whose names start with a, except indexes whose names start with an.

Parameter	Description
Index Deletion	<ul style="list-style-type: none"> ◦ Index Names Only: When you delete an index, you must specify its name. ◦ Allow Wildcard Characters: You can use wildcards to delete multiple indexes at a time. <div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p> Notice You cannot restore the indexes that have been deleted. Exercise caution when you specify this parameter.</p> </div>
Other Configurations	<p>The following content lists some supported configuration items. For more information, see YML configuration parameters.</p> <ul style="list-style-type: none"> ◦ http.cors.enabled ◦ http.cors.allow-origin ◦ http.cors.max-age ◦ http.cors.allow-methods ◦ http.cors.allow-headers ◦ http.cors.allow-credentials ◦ reindex.remote.whitelist ◦ action.auto_create_index ◦ action.destructive_requires_name <div style="background-color: #e0f2f1; padding: 10px; border: 1px solid #ccc;"> <p> Notice You can reindex data from a remote Elasticsearch cluster that is specified in a remote reindex whitelist. You can reindex data from all versions of remote Elasticsearch clusters. This feature allows you to migrate index data from an Elasticsearch cluster of an earlier version to an Elasticsearch cluster of the newly released version. For more information, see Reindex data from a remote Elasticsearch cluster.</p> </div>

10.3.6.2.2. YML configuration parameters

This topic describes the YML configuration parameters supported by Elasticsearch. These configuration parameters support only the static configuration mode but not the hot deployment mode. For configurations to take effect, you must add the configurations to the `elasticsearch.yml` file.

Parameters

Parameter	Description
-----------	-------------

Parameter	Description
http.cors.enabled	<p>The cross-origin resource sharing (CORS) configuration item. This item is used to specify whether to allow browsers on other origins to access Elasticsearch.</p> <ul style="list-style-type: none"> • If you set the value to <code>true</code>, CORS is enabled, and Elasticsearch can process <code>OPTIONS CORS</code> requests. • If the origin in a request is declared in <code>http.cors.allow-origin</code>, Elasticsearch returns a response that has the <code>Access-Control-Allow-Origin</code> header included. • The default value is <code>false</code>. If you set the value to <code>false</code>, CORS is disabled. In this case, Elasticsearch ignores the origin in the request header and returns a response that does not have the <code>Access-Control-Allow-Origin</code> header included. • If a client cannot send a pre-flight request that has origin information included in the request header or does not verify the <code>Access-Control-Allow-Origin</code> header in the response that is returned from the server, the cross-origin security is compromised. • If CORS is disabled for Elasticsearch, a client can send only an <code>OPTIONS</code> request to check whether the <code>Access-Control-Allow-Origin</code> header exists.
http.cors.allow-origin	<p>The origin configuration item. This item specifies the origins from which requests are allowed. By default, no origins are allowed.</p> <ul style="list-style-type: none"> • If you add a forward slash (<code>/</code>) to the start and end of the value, this item is considered as a regular expression. • This allows you to use regular expressions to support HTTP and HTTPS requests. For example, <code>/https?:\/\/localhost(:[0-9]+)?/</code> indicates that Elasticsearch responds to all requests that match the regular expression. • The asterisk (<code>*</code>) is a valid character but considered as a security risk. This is because the asterisk indicates that an Elasticsearch cluster is open to all origins. We recommend that you do not use asterisks.
http.cors.max-age	<p>Browsers can send <code>OPTIONS</code> requests to query the CORS configuration. This item specifies the cache time of the retrieved CORS configuration. The default value is 1728000 seconds (20 days).</p>
http.cors.allow-methods	<p>The item that is used to configure the request method. Valid values: <code>OPTIONS, HEAD, GET, POST, PUT, and DELETE</code>.</p>
http.cors.allow-headers	<p>The item that is used to configure the request header. Valid values: <code>X-Requested-With, Content-Type, and Content-Length</code>.</p>
http.cors.allow-credentials	<p>The credential configuration item. This item specifies whether Elasticsearch is allowed to return the <code>Access-Control-Allow-Credentials</code> header. If you set the value to <code>true</code>, Elasticsearch is allowed to return the header. The default value is <code>false</code>.</p>

Parameter	Description
<code>reindex.remote.whitelist</code>	The remote reindex whitelist of the current cluster. This whitelist contains the addresses of remote hosts that can be used to access the current cluster. An address in the whitelist can be a combination of host and port. Separate the configurations of multiple hosts with commas (,), such as <code>otherhost:9200,another:9200,127.0.10.**:9200,localhost:**</code> . You can use only the host and port to configure security policies because the whitelist ignores the protocol information.
<code>action.auto_create_index</code>	The configuration item for auto index creation. If you set the value to <code>false</code> , the auto index creation feature is disabled.
<code>action.destructive_requires_name</code>	This item specifies whether to specify the name of an index when you delete the index. The default value is <code>false</code> . If you set the value to <code>false</code> , you can use a regular expression or the <code>_all</code> parameter to delete indexes. If you set the value to <code>true</code> , you must specify the names of the indexes you want to delete. In this case, you cannot use <code>_all</code> or wildcards.

10.3.6.2.3. Reindex data from a remote Elasticsearch cluster

This topic describes how to reindex data from a remote Elasticsearch cluster that is specified in a remote reindex whitelist. This feature is applicable to all versions of remote Elasticsearch clusters. It allows you to reindex data from an Elasticsearch cluster of an earlier version to an Elasticsearch cluster of the newly released version.

You can call the reindex operation to reindex data. Example:

```
POST _reindex
{
  "source": {
    "remote": {
      "host": "http://otherhost:9200",
      "username": "user",
      "password": "pass"
    },
    "index": "source",
    "query": {
      "match": {
        "test": "data"
      }
    }
  },
  "dest": {
    "index": "dest"
  }
}
```

 **Note** Reindexing from a remote cluster does not support manual slicing or automatic slicing.

- The value of the `host` parameter must include the protocol, domain name, and port number, such as `http://otherhost:9200`.
- `username` and `password` are optional. If the remote Elasticsearch cluster needs to use the basic authorization scheme, specify the username and password. If you use the basic authorization scheme, we

recommend that you use the HTTPS protocol. Otherwise, the password is transmitted as a text.

- The reindex operation can be called remotely only after the remote host address is declared in the `elasticsearch.yml` configuration file by using the `reindex.remote.whitelist` attribute. `reindex.remote.whitelist` can use host-port pairs. Separate multiple pairs with commas (,), such as, `otherhost:9200, another:9200, 127.0.10.**:9200, localhost:**`. You can use only the host and port to configure security policies because the whitelist ignores the protocol information.
- If the host is already added to the whitelist, the query is directly sent to the remote Elasticsearch cluster without verification or modification.

Indexing from a remote Elasticsearch cluster uses on-heap buffer. The default maximum batch size is 100 MB. If the index on the remote cluster contains large documents, you must adjust the batch size to a small value.

In the following example, the batch size is 10, which is the minimum value.

```
POST _reindex
{
  "source": {
    "remote": {
      "host": "http://otherhost:9200"
    },
    "index": "source",
    "size": 10,
    "query": {
      "match": {
        "test": "data"
      }
    }
  },
  "dest": {
    "index": "dest"
  }
}
```

- `socket_timeout` is used to set the timeout period for socket read. The default value is 30s.
- `connect_timeout` is used to set the connection timeout period. The default value is 1s.

In the following example, `socket_timeout` is set to 1m and `connect_timeout` is set to 10s.

```

POST _reindex
{
  "source": {
    "remote": {
      "host": "http://otherhost:9200",
      "socket_timeout": "1m",
      "connect_timeout": "10s"
    },
    "index": "source",
    "query": {
      "match": {
        "test": "data"
      }
    }
  },
  "dest": {
    "index": "dest"
  }
}

```

10.3.7. Configure plug-ins

Plug-ins extend the capabilities of Apsara Stack Elasticsearch in data pre-processing and analytics. You can use built-in plug-ins or upload custom plug-ins based on your requirements. This topic describes how to configure a plug-in.

Go to the plug-in configuration page

1. [Log on to the Elasticsearch console.](#)
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane, click **Plug-ins**.

Built-in plug-ins

The following table lists the built-in plug-ins for Elasticsearch on the Plug-ins page.

Plug-in	Description
analysis-icu	The ICU analysis plug-in for Elasticsearch. This plug-in integrates the ICU module of Lucene into Elasticsearch and adds ICU analysis components.
analysis-ik	The IK analysis plug-in for Elasticsearch.
analysis-kuromoji	The Japanese (Kuromoji) analysis plug-in for Elasticsearch. This plug-in integrates the Kuromoji analysis module of Lucene into Elasticsearch.
analysis-phonetic	The phonetic analysis plug-in for Elasticsearch. This plug-in integrates the phonetic token filter into Elasticsearch.
analysis-pinyin	The pinyin analysis plug-in for Elasticsearch.
analysis-smartcn	The smart Chinese analysis plug-in for Elasticsearch. This plug-in integrates the smart Chinese analysis module of Lucene into Elasticsearch.

Plug-in	Description
elasticsearch-repository-oss	The plug-in allows you to use Apsara Stack Object Storage Service (OSS) to store Elasticsearch snapshots.
gab-paas-plugin	None.
ingest-attachment	The ingest processor for Elasticsearch. This plug-in uses Apache Tika to extract content.
ingest-geoip	The ingest processor for Elasticsearch. This plug-in queries geographic data in MaxMind databases based on IP addresses.
ingest-user-agent	The ingest processor for Elasticsearch. This plug-in extracts information from a user agent.
mapper-murmur3	When you create an index and store these values in the index, this plug-in allows you to compute the hash values of fields.
mapper-size	This plug-in allows you to take note of the sizes of documents before they are compressed when you create an index.
repository-hdfs	This plug-in provides support for Hadoop Distributed File System (HDFS) repositories.
search-guard-6	This plug-in provides the access control feature for Elasticsearch.

The analysis-ik plug-in allows you to use the standard update or rolling update method to update IK dictionaries. For more information, see [Standard update of IK dictionaries](#) and [Rolling update of IK dictionaries](#).

Standard update of IK dictionaries

The standard update method requires Elasticsearch to perform a rolling restart for an Elasticsearch cluster to update dictionaries. Elasticsearch sends the uploaded dictionary file to all nodes in the cluster, modifies the `IKAnalyzer.cfg.xml` file, and then restarts the nodes to load the file.

You can use the standard update method to update the IK main dictionary and stopword list. In the standard update pane, you can check the built-in main dictionary `SYSTEM_MAIN.dic` and the built-in stopword list `SYSTEM_STOPWORD.dic`.

- If you want to update the built-in main dictionary, upload a dictionary file named `SYSTEM_MAIN.dic`.
 - If you want to update the built-in stopword list, upload a dictionary file named `SYSTEM_STOPWORD.dic`.
1. [Go to the plug-in configuration page](#).
 2. On the page that appears, find the analysis-ik plug-in and click **Standard Update** in the Actions column.
 3. In the **Plug-ins** pane, click **Configure**.
 4. Click **Upload DIC File** and select a local file to upload.

Note

- By default, a .dic file needs to be uploaded. You can also choose to upload an OSS file.
- If the content of a dictionary stored in OSS changes, you must manually upload the dictionary file again.

5. In the lower part of the pane, select **This operation will restart the instance. Continue?** and click **Save**.

Then, the system performs a rolling start for the Elasticsearch cluster.

6. After the cluster is restarted, log on to the Kibana console of the Elasticsearch cluster and run the following command to check whether the new dictionaries take effect.

For more information about how to log on to the Kibana console, see [Use the Kibana console to access an Elasticsearch cluster](#).

```
GET _analyze
{
  "analyzer": "ik_smart",
  "text": ["Tokens in the new dictionaries"]
}
```

Note

- You cannot delete the built-in main dictionary and stopword list.
- The standard update operation requires Elasticsearch to perform a rolling restart for the cluster.
- You can perform the standard update operation only when the cluster is healthy.

Rolling update of IK dictionaries

If the content of a dictionary file changes, you can use this method to update the dictionaries on all nodes in an Elasticsearch cluster. After you upload the latest dictionary file, all nodes in the Elasticsearch cluster automatically load the file.

If the dictionary file list changes when you perform a rolling update, the system performs a rolling restart for your Elasticsearch cluster to reload the dictionary configuration. For example, when you upload a new dictionary file or delete an existing dictionary file, the changes are synchronized to the `IKAnalyzer.cfg.xml` file.

The procedure of a rolling update is similar to that of a standard update. If you upload a dictionary file for the first time, you must modify the `IKAnalyzer.cfg.xml` file. Elasticsearch needs to perform a rolling restart to reload the configuration file. Subsequently, if you upload a dictionary file with the same name, Elasticsearch does not perform a rolling restart for the update to take effect.

1. [Go to the plug-in configuration page](#).
2. On the page that appears, find the analysis-ik plug-in and click **Rolling Update** in the Actions column.
3. In the **Plug-ins** pane, click **Configure**.
4. Click **Upload DIC File** and select a local file to upload.

Note

- By default, a .dic file needs to be uploaded. You can also choose to upload an OSS file.
- If the content of a dictionary stored in OSS changes, you must manually upload the dictionary file again.

5. In the lower part of the pane, select **This operation will restart the instance. Continue?** and click **Save**.

After you click Save, the system performs a rolling restart for the cluster. After the cluster is restarted, the new dictionaries take effect.

If you want to add tokens to or remove tokens from the new dictionaries, perform the following steps to modify the `a_10words.dic` file: In the rolling update pane, delete the existing `a_10words.dic` file and upload a new dictionary file. The new dictionary file must have the same name. This operation changes the content of the existing dictionary file in the cluster and uploads a new file that has the same name. Elasticsearch does not need to perform a rolling restart on the cluster for the update to take effect. You can directly click **Save**.

The analysis-ik plug-in on the nodes of the Elasticsearch cluster automatically loads the dictionary file. The time that is required to load the dictionary file varies depending on the nodes. It requires about two minutes for all nodes to load the dictionary file. To verify the new dictionaries, you can log on to the Kibana console of the Elasticsearch cluster and run the following command multiple times:

```
GET _analyze
{
  "analyzer": "ik_smart",
  "text": ["Tokens in the new dictionaries"]
}
```

 **Note** You cannot use the rolling update method to modify the built-in main dictionary. If you want to modify the built-in main dictionary, you must use the standard update method.

10.3.8. Configure security settings

You can reset the password of your Elasticsearch cluster, modify the Kibana whitelist, and modify the VPC whitelist. This topic describes how to configure security settings.

Go to the security configuration page

1. [Log on to the Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane, click **Security**.

Reset the password of an Elasticsearch cluster

 **Notice** This operation only resets the password of the elastic account. We recommend that you do not use the elastic account to log on to your Elasticsearch cluster.

1. [Go to the security configuration page](#).
2. Click **Reset** next to **Elasticsearch Instance Password**.
3. In the **Reset** pane, enter a new password and confirm the password.
4. Click **OK**.

Then, the new password takes effect within about five minutes. The new password will be used to log on to the Kibana console and access the Elasticsearch cluster.

 **Note** The password reset does not restart the Elasticsearch cluster.

Modify a Kibana whitelist

1. [Go to the security configuration page](#).
2. Click **Update** next to **Kibana Whitelist**.
3. In the **Edit Kibana Whitelist** pane, enter IP addresses.

You can enter both IP addresses and CIDR blocks. For example, enter `192.168.0.1` or `192.168.0.0/24`. Separate multiple IP addresses or CIDR blocks with commas (.). You can enter `127.0.0.1` to deny requests from all IPv4 addresses or enter `0.0.0.0/0` to allow requests from all IPv4 addresses.

If your Elasticsearch cluster is deployed in the China (Hangzhou) region, you can add IPv6 addresses to the whitelist. For example, enter `2401:b180:1000:24::5` or `2401:b180:1000::/48`. Enter `::1` to deny requests from all IPv6 addresses or `::/0` to allow requests from all IPv6 addresses.

 **Note** You can use only a server that resides in the same VPC as your Elasticsearch cluster to log on to the Kibana console.

4. Click **OK**.

Modify a VPC whitelist

1. [Go to the security configuration page](#).
2. Click **Update** next to **VPC Whitelist**.
3. In the **Edit VPC Whitelist** pane, enter IP addresses.

You can enter both IP addresses and CIDR blocks. For example, enter `192.168.0.1` or `192.168.0.0/24`. Separate multiple IP addresses or CIDR blocks with commas (.). You can enter `127.0.0.1` to deny requests from all IPv4 addresses or enter `0.0.0.0/0` to allow requests from all IPv4 addresses.

 **Note**

- By default, requests from all IPv4 addresses within the VPC in which the Elasticsearch cluster resides are allowed.
- The VPC whitelist is used to control access from IP addresses within VPCs.

4. Click **OK**.

10.3.9. Back up data

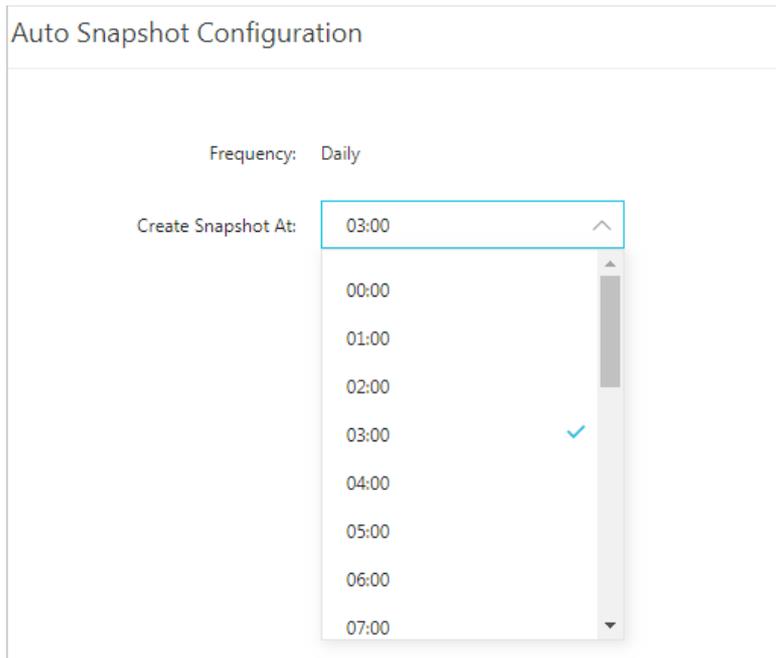
10.3.9.1. Enable and configure the auto snapshot feature

The auto snapshot feature allows you to specify the snapshot creation period and time. After you specify them, the system automatically creates snapshots. This topic describes how to enable and configure the auto snapshot feature.

1. [Log on to the Elasticsearch console](#).
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane, click **Snapshots**.
4. Turn on the **Auto Snapshot** switch. This switch is turned on by default.

 **Notice** After the auto snapshot feature is enabled, the system uses the system time of the region where your Elasticsearch cluster resides to create snapshots. Do not perform snapshot operations when the system is creating snapshots.

5. Click **Modify Configuration** on the right side of **Snapshots (Free Trial)**.
6. In the **Auto Snapshot Configuration** pane, set **Create Snapshot At**.



- **Frequency:** The value of this parameter is Daily.
- **Create Snapshot At :** specifies the hour for creating a snapshot every day. Valid values: 00:00 to 23:00.

7. Click Save.

10.3.9.2. Query snapshot status

After you enable the auto snapshot feature, you can call the snapshot operation to query the status of snapshots. This topic describes how to query the status of snapshots.

Note You can run all the commands provided in this topic in the Kibana console. For information about how to log on to the Kibana console, see [Access an Elasticsearch cluster](#).

Query all snapshots

Run the following command to query information about all snapshots that are stored in the `aliyun_auto_snapshot` repository:

```
GET _snapshot/aliyun_auto_snapshot/_all
```

If the command is successfully executed, the following result is returned:

```
{
  "snapshots": [
    {
      "snapshot": "<yourSnapshotName>",
      "uuid": "n7YlavyZTm2hwg8BeW****",
      "version_id": 5050399,
      "version": "5.5.3",
      "indices": [
        ".kibana"
      ],
      "state": "SUCCESS",
      "start_time": "2018-06-28T01:22:39.609Z",
      "start_time_in_millis": 1530148959609,
      "end_time": "2018-06-28T01:22:39.923Z",
      "end_time_in_millis": 1530148959923,
      "duration_in_millis": 314,
      "failures": [],
      "shards": {
        "total": 1,
        "failed": 0,
        "successful": 1
      }
    },
    {
      "snapshot": "<yourSnapshotName>",
      "uuid": "frdl1YFzQ5Cn5xN9ZW****",
      "version_id": 5050399,
      "version": "5.5.3",
      "indices": [
        ".kibana"
      ],
      "state": "SUCCESS",
      "start_time": "2018-06-28T01:25:00.764Z",
      "start_time_in_millis": 1530149100764,
      "end_time": "2018-06-28T01:25:01.482Z",
      "end_time_in_millis": 1530149101482,
      "duration_in_millis": 718,
      "failures": [],
      "shards": {
        "total": 1,
        "failed": 0,
        "successful": 1
      }
    }
  ]
}
```

The state field indicates the status of a snapshot. A snapshot can be in one of the following states:

- **IN_PROGRESS** : The snapshot is being created.
- **SUCCESS** : The snapshot is created, and all shards are stored.
- **FAILED** : The snapshot fails to be created because some shards cannot be stored.
- **PARTIAL** : The snapshot is created, but a minimum of one shard fails to be stored.
- **INCOMPATIBLE** : The snapshot is incompatible with the Elasticsearch cluster version.

Query a specified snapshot

Run the following command to query information about a specified snapshot that is stored in the `aliyun_auto_snapshot` repository:

```
GET _snapshot/aliyun_auto_snapshot/<snapshot>/_status
```

Replace `<snapshot>` with the name of the snapshot, such as `<yourSnapshotName>`.

If the command is successfully executed, the following result is returned:

```
{
  "snapshots": [
    {
      "snapshot": "<yourSnapshotName>",
      "repository": "aliyun_auto_snapshot",
      "uuid": "n7YlayyZTm2hwg8BeWbydA",
      "state": "SUCCESS",
      "shards_stats": {
        "initializing": 0,
        "started": 0,
        "finalizing": 0,
        "done": 1,
        "failed": 0,
        "total": 1
      },
      "stats": {
        "number_of_files": 4,
        "processed_files": 4,
        "total_size_in_bytes": 3296,
        "processed_size_in_bytes": 3296,
        "start_time_in_millis": 1530148959688,
        "time_in_millis": 77
      },
      "indices": {
        ".kibana": {
          "shards_stats": {
            "initializing": 0,
            "started": 0,
            "finalizing": 0,
            "done": 1,
            "failed": 0,
            "total": 1
          },
          "stats": {
            "number_of_files": 4,
            "processed_files": 4,
            "total_size_in_bytes": 3296,
            "processed_size_in_bytes": 3296,
            "start_time_in_millis": 1530148959688,
            "time_in_millis": 77
          },
          "shards": {
            "0": {
              "stage": "DONE",
              "stats": {
                "number_of_files": 4,
                "processed_files": 4
              }
            }
          }
        }
      }
    }
  ]
}
```

```

      "processed_size_in_bytes": 3296,
      "start_time_in_millis": 1530148959688,
      "time_in_millis": 77
    }
  }
}
]
}

```

10.3.9.3. Restore data from automatic snapshots

If you enable the auto snapshot feature for an Elasticsearch cluster, the system creates snapshots for the cluster every day. You can call the restore operation to restore data to the cluster from the created snapshots. This topic describes how to restore data from automatic snapshots.

 **Note** You can run all the commands provided in this topic in the Kibana console. For information about how to log on to the Kibana console, see [Access an Elasticsearch cluster](#).

Background information

- The first snapshot is a full copy of the data in an Elasticsearch cluster. Subsequent snapshots store only incremental data. Therefore, it requires longer time to create the first snapshot than a subsequent snapshot.
- The system stores automatic snapshots that are created only within the last five days.
- A snapshot does not store monitoring data generated by an Elasticsearch cluster, such as indexes with the prefix `.monitoring` or `.security_audit`.
- An automatic snapshot repository is created when the first snapshot is created.

Query all snapshot repositories

Run the following command to query all snapshot repositories:

```
GET _snapshot
```

If the command is successfully executed, the following result is returned:

```

{
  "aliyun_auto_snapshot": {
    "type": "oss",
    "settings": {
      "compress": "true",
      "base_path": "xxx",
      "endpoint": "xxx"
    }
  }
}

```

- `aliyun_auto_snapshot`: the name of the repository.

- `type`: the storage where snapshots are stored.
- `compress`: indicates whether compression is enabled. The value `true` indicates that the metadata of indexes is compressed during snapshot creation.
- `base_path`: the location of snapshots in Object Storage Service (OSS).
- `endpoint`: the endpoint of the OSS bucket.

The auto snapshot feature also supports the following default parameters that are not displayed:

- `max_snapshot_bytes_per_sec:40mb` : The maximum speed for snapshot creation on a single node is 40 MB/s.
- `max_restore_bytes_per_sec:40mb` : The maximum speed for data restoration on a single node is 40 MB/s.
- `chunk_size: Max 1Gb` : During snapshot creation, a large index is divided into multiple parts. The maximum size of each part is 1 GB.

Query all snapshots

Run the following command to query information about all snapshots that are stored in the `aliyun_auto_snapshot` repository:

```
GET _snapshot/aliyun_auto_snapshot/_all
```

If the command is successfully executed, the following result is returned:

```
{
  "snapshots": [
    {
      "snapshot": "<yourSnapshotName>",
      "uuid": "MMRniVLPRaiawSCm8D****",
      "version_id": 5050399,
      "version": "5.5.3",
      "indices": [
        "index_1",
        ".security",
        ".kibana"
      ],
      "state": "SUCCESS",
      "start_time": "2018-06-27T01:16:01.009Z",
      "start_time_in_millis": 1530062161009,
      "end_time": "2018-06-27T01:16:05.632Z",
      "end_time_in_millis": 1530062165632,
      "duration_in_millis": 4623,
      "failures": [],
      "shards": {
        "total": 12,
        "failed": 0,
        "successful": 12
      }
    }
  ]
}
```

Restore indexes from a snapshot

You can call the `_restore` operation to restore indexes from snapshots.

- Run the following command to restore all indexes from a specified snapshot that is stored in the

aliyun_auto_snapshot repository. The restoration task is executed at the backend.

```
POST _snapshot/aliyun_auto_snapshot/<snapshot>/_restore
```

Replace `<snapshot>` with the name of the snapshot, such as `<yourSnapshotName>`.

- Run the following command to restore all indexes from a specified snapshot that is stored in the aliyun_auto_snapshot repository. Then, wait until the restoration task is complete.

The `_restore` operation runs restoration tasks asynchronously. An Elasticsearch cluster immediately returns a response if the `_restore` operation can be executed. The restoration task is executed at the backend. You can specify the `wait_for_completion` parameter to enable the cluster to return a response only after the restoration task is complete.

```
POST _snapshot/aliyun_auto_snapshot/<snapshot>/_restore?wait_for_completion=true
```

Replace `<snapshot>` with the name of the snapshot, such as `<yourSnapshotName>`.

- Run the following command to restore specified indexes from a specific snapshot that is stored in the aliyun_auto_snapshot repository, and rename the restored indexes. The restoration task is executed at the backend.

```
POST _snapshot/aliyun_auto_snapshot/<snapshot>/_restore
{
  "indices": "index_1",
  "rename_pattern": "index_(.+)",
  "rename_replacement": "restored_index_$1"
}
```

- `<snapshot>`: Replace it with the name of the snapshot, such as `<yourSnapshotName>`.
- `indices`: the name of the index you want to restore.
- `rename_pattern`: optional. This parameter specifies the regular expression that is used to match the name of the index you want to restore.
- `rename_replacement`: optional. This parameter specifies the regular expression that is used to rename a matched index.

10.3.9.4. Commands for creating snapshots and restoring data

You can call the snapshot operation to back up or restore data for your Apsara Stack Elasticsearch cluster. The snapshot operation retrieves the status and data of your cluster and then stores them to a shared repository.

The first snapshot is a full copy of the data in a cluster. Subsequent snapshots store only incremental data. Therefore, when you create subsequent snapshots, the system only needs to add data to or remove data from the snapshots. This means that it requires less time to create a subsequent snapshot than the first snapshot.

Precautions

- This topic uses the following markers to provide descriptions for code: `<1>`, `<2>`, and `<3>`. Remove these markers when you run the code.
- You can run the code provided in this topic in the Kibana console of your Elasticsearch cluster. For more information, see [Access an Elasticsearch cluster](#).

Create a repository

```
PUT _snapshot/my_backup
{
  "type": "oss",
  "settings": {
    "endpoint": "http://oss-cn-hangzhou-internal.aliyuncs.com", <1>
    "access_key_id": "xxxx",
    "secret_access_key": "xxxxxx",
    "bucket": "xxxxxx", <2>
    "compress": true,
    "base_path": "snapshot/" <3>
  }
}
```

- <1>: The `endpoint` parameter specifies the internal endpoint of the OSS bucket.
- <2>: The `bucket` parameter specifies the name of an OSS bucket that has been created.
- <3>: The `base_path` parameter specifies the path of the repository. The default value is the root directory.

Set the size of each part

When you upload large volumes of data to an OSS bucket, you can use the `chunk_size` parameter to set the size of each part. This allows you to upload the data in multiple parts.

```
POST _snapshot/my_backup/ <1>
{
  "type": "oss",
  "settings": {
    "endpoint": "http://oss-cn-hangzhou-internal.aliyuncs.com",
    "access_key_id": "xxxx",
    "secret_access_key": "xxxxxx",
    "bucket": "xxxxxx",
    "chunk_size": "500mb",
    "base_path": "snapshot/" <2>
  }
}
```

- <1>: Use the POST method instead of the PUT method. The POST method updates repository settings.
- <2>: The `base_path` parameter specifies the path of the repository. The default value is the root directory.

Query repository information

```
GET _snapshot
```

You can also use the `GET _snapshot/my_backup` command to query the information of a specified repository.

Create snapshots

The following command is a basic command that is used to create a snapshot:

```
PUT _snapshot/my_backup/snapshot_1
```

This command creates the `snapshot_1` snapshot for all open indexes. The snapshot is stored in the `my_backup` repository. After you run the command, the system immediately returns a response while the snapshot is created at the backend.

If you want the system to return a response after it creates the snapshot, add the `wait_for_completion` parameter to the command.

```
PUT _snapshot/my_backup/snapshot_1? wait_for_completion=true
```

After you run the command, the system does not return a response until the snapshot is created. If the size of the index is large, the response is returned after a longer period of time.

Create a snapshot for specified indexes

By default, a snapshot contains all open indexes. For Kibana, when you create a snapshot, you may want to ignore all diagnostic indexes (the `.kibana` indexes) because of limited disk space. To create a snapshot for specified indexes, run the following command:

 **Notice** A repository stores multiple snapshots. Each snapshot is a copy of all indexes, specified indexes, or a single index in a cluster. When you create a snapshot, make sure that the snapshot name is unique.

```
PUT _snapshot/my_backup/snapshot_2
{
  "indices": "index_1,index_2"
}
```

The preceding command creates a snapshot only for the `index1` and `index2` indexes.

Query snapshot information

In some cases, you may need to query snapshot information. For example, a snapshot name containing a date is hard to remember, such as `backup_2014_10_28`.

To query the information of a snapshot, send a `GET` request that contains both the repository name and snapshot name. Example:

```
GET _snapshot/my_backup/snapshot_2
```

The following response contains detailed information of the snapshot:

```
{
  "snapshots": [
    {
      "snapshot": "snapshot_2",
      "indices": [
        ".marvel_2014_28_10",
        "index1",
        "index2"
      ],
      "state": "SUCCESS",
      "start_time": "2014-09-02T13:01:43.115Z",
      "start_time_in_millis": 1409662903115,
      "end_time": "2014-09-02T13:01:43.439Z",
      "end_time_in_millis": 1409662903439,
      "duration_in_millis": 324,
      "failures": [],
      "shards": {
        "total": 10,
        "failed": 0,
        "successful": 10
      }
    }
  ]
}
```

You can replace the snapshot name in the preceding command with `_all` to query all snapshots in the repository. Example:

```
GET _snapshot/my_backup/_all
```

Delete a snapshot

You can specify a repository name and a snapshot name, and send a `DELETE` request to delete the specified snapshot. Example:

```
DELETE _snapshot/my_backup/snapshot_2
```

Notice

- You can only use the `DELETE` operation to delete snapshots. A snapshot is associated with other backup files. Some of the files may also be used by other snapshots. The `DELETE` operation does not delete files that are still being used by other snapshots. It only deletes files that are associated with the deleted snapshot and are not used by other snapshots.
- If you choose to manually delete a snapshot, you may delete files that are associated with snapshots by mistake. This may cause data loss.

Monitor snapshot creation progress

The `wait_for_completion` parameter provides a simple method for you to monitor the progress of a snapshot creation task. However, this parameter is not suitable for snapshot creation tasks of medium-size Elasticsearch clusters. You can use one of the following methods to query detailed information about a snapshot:

- Send a `GET` request with the snapshot name specified. Example:

```
GET _snapshot/my_backup/snapshot_3
```

If the system is still creating the snapshot when you run the preceding command, the information of the creation task is returned, such as the time when the snapshot creation task started and the duration.

 **Notice** The preceding command shares a thread pool with the command used to create a snapshot. Therefore, if you create a snapshot for large shards, the preceding command has to wait until the resources that are used by the snapshot creation command in the thread pool are released.

- Call the `_status` operation to query the snapshot status.

```
{
  "snapshots": [
    {
      "snapshot": "snapshot_3",
      "repository": "my_backup",
      "state": "IN_PROGRESS", <1>
      "shards_stats": {
        "initializing": 0,
        "started": 1, <2>
        "finalizing": 0,
        "done": 4,
        "failed": 0,
        "total": 5
      },
      "stats": {
        "number_of_files": 5,
        "processed_files": 5,
        "total_size_in_bytes": 1792,
        "processed_size_in_bytes": 1792,
        "start_time_in_millis": 1409663054859,
        "time_in_millis": 64
      },
      "indices": {
        "index_3": {
          "shards_stats": {
            "initializing": 0,
            "started": 0,
            "finalizing": 0,
            "done": 5,
            "failed": 0,
            "total": 5
          },
          "stats": {
            "number_of_files": 5,
            "processed_files": 5,
            "total_size_in_bytes": 1792,
            "processed_size_in_bytes": 1792,
            "start_time_in_millis": 1409663054859,
            "time_in_millis": 64
          },
          "shards": {
            "0": {
              "stage": "DONE",
              "stats": {
                "number_of_files": 1,
                "processed_files": 1,
                "total_size_in_bytes": 514,
                "processed_size_in_bytes": 514,
                "start_time_in_millis": 1409663054862,
                "time_in_millis": 22
              }
            }
          }
        }
      },
      ...
    }
  ]
}
```

- <1>: The status of the snapshot. If a snapshot is being created, the value of the field is `IN_PROGRESS`.

- `<2>`: The number of shards that are being transmitted. If the value 1 is returned, a shard of the snapshot is being transmitted, and the other four shards have been transmitted.

The value of the `shards_stats` parameter contains the status of the snapshot. It also contains statistics about each index and shard. This parameter allows you to learn the detailed information of the snapshot creation progress. A shard can be in one of the following states:

- `INITIALIZING` : The shard is verifying the status of the cluster to check whether the shard can be stored in a snapshot. In most cases, this process is fast.
- `STARTED` : Data is being transmitted to the repository.
- `FINALIZING` : The data transmission process is complete. The shard is sending snapshot metadata.
- `DONE` : The snapshot is created.
- `FAILED` : An error occurred during the snapshot creation. The shard, index, or snapshot cannot be processed. You can view logs for more information.

Cancel a snapshot

To cancel a snapshot, run the following command when the snapshot is being created:

```
DELETE _snapshot/my_backup/snapshot_3
```

This command stops the snapshot creation process and deletes the snapshot that is being created from the repository.

Restore indexes from a snapshot

To restore indexes from a snapshot, run the command that is used in the ["Create a repository"](#) section on the Elasticsearch cluster that stores these indexes. You can use one of the following methods to restore indexes from a snapshot:

- To restore indexes from a specified snapshot, append the `_restore` parameter to the snapshot name in the command to run. Example:

```
POST _snapshot/my_backup/snapshot_1/_restore
```

After you run this command, the system restores all indexes in the snapshot. For example, if the `snapshot_1` snapshot contains five indexes, all these indexes are restored to the Elasticsearch cluster. You can also specify the indexes that you want to restore. For more information, see [Create a snapshot for specified indexes](#).

- Restore specified indexes and rename the indexes. If you only want to verify or process the data in indexes and do not need to overwrite the data, use this method to restore the indexes.

```
POST /_snapshot/my_backup/snapshot_1/_restore
{
  "indices": "index_1", <1>
  "rename_pattern": "index_(.+)", <2>
  "rename_replacement": "restored_index_$1" <3>
}
```

In this example, the `index_1` index is restored to your Elasticsearch cluster and renamed `restored_index_1`.

- `<1>`: The system only restores the `index_1` index from the snapshot.
 - `<2>`: The system searches for the index that is being restored and matches the index name with the provided pattern.
 - `<3>`: The system renames the matching index.
- If you want the system to return a response after it restores the index, add the `wait_for_completion`

parameter to the command.

```
POST _snapshot/my_backup/snapshot_1/_restore?wait_for_completion=true
```

After you call the `_restore` operation, the system immediately returns a response and restores the index at the backend. If you want the operation to return the result after the restore process is complete, add the `wait_for_completion` parameter.

Monitor index restoration progress

Note Restoring data from a repository applies the existing restoration mechanism in Elasticsearch. Restoring shards from a repository is the same as restoring data from a node.

You can call the recovery operation to monitor the progress of an index restoration task.

- Monitor a specified index that is being restored.

```
GET restored_index_3/_recovery
```

The recovery operation is a general-purpose operation. It shows the status of the shards that are being transmitted to your cluster.

- Monitor all indexes on the cluster. This may include shards that are irrelevant to the restoration process.

```
GET /_recovery/
```

The following content shows the sample response:

```

{
  "restored_index_3": {
    "shards": [ {
      "id": 0,
      "type": "snapshot", <1>
      "stage": "index",
      "primary": true,
      "start_time": "2014-02-24T12:15:59.716",
      "stop_time": 0,
      "total_time_in_millis": 175576,
      "source": { <2>
        "repository": "my_backup",
        "snapshot": "snapshot_3",
        "index": "restored_index_3"
      },
      "target": {
        "id": "ryqJ5lO5S4-ISFbGntkEkg",
        "hostname": "my.fqdn",
        "ip": "10.0.**.**",
        "name": "my_es_node"
      },
      "index": {
        "files": {
          "total": 73,
          "reused": 0,
          "recovered": 69,
          "percent": "94.5%" <3>
        },
        "bytes": {
          "total": 79063092,
          "reused": 0,
          "recovered": 68891939,
          "percent": "87.1%"
        },
        "total_time_in_millis": 0
      },
      "translog": {
        "recovered": 0,
        "total_time_in_millis": 0
      },
      "start": {
        "check_index_time": 0,
        "total_time_in_millis": 0
      }
    }
  ]
}

```

- <1>: The `type` parameter indicates the type of restoration. The value `snapshot` indicates that the shard is being restored from a snapshot.
- <2>: The `source` parameter indicates the source snapshot and repository.
- <3>: The `percent` parameter indicates the progress of the restoration task. The value `94.5%` indicates that 94.5% of the shard files are restored.

The response lists all indexes that are being restored and the shards in these indexes. Each shard has statistics about the start or end time, duration, restoration progress, and bytes transmitted.

Cancel index restoration

To cancel index restoration, you only need to delete the indexes that are being restored. A restoration process is a shard restoration process. You can call the DELETE operation to modify the status of the cluster and cancel the restoration process. Example:

```
DELETE /restored_index_3
```

If you run the preceding command when the `restored_index_3` index is being restored, the system stops the restoration and deletes the data that has been restored to the cluster. For more information, see [Snapshot And Restore](#).

Use a snapshot to migrate data

To use a snapshot to migrate data from an Elasticsearch cluster to another, follow these steps:

1. Back up a snapshot to OSS.
2. Create a snapshot repository on the destination cluster. The repository must use the OSS bucket that stores the snapshot.
3. Set the `base_path` parameter to the path of the snapshot.
4. Run the data restoration command on the destination cluster.

 **Note** These steps provide a simple solution to snapshot-based data migration. For more information, see [Use a snapshot stored in OSS to migrate Elasticsearch data](#).

10.4. Manage documents

10.4.1. Create a document

This topic describes how to use Elasticsearch to create a document.

[Access an Elasticsearch cluster](#) and send a POST request to create a document in the cluster.

```
curl -XPOST http://<HOST>:<PORT>/my_index/my_type -d '{"title": "One", "tags": ["ruby"]}'
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is 9200.
- `my_index` : the name of the index, which can be customized.
- `my_type` : the type of the document, which can be customized.

 **Note**

- Each document has an ID and a type. The response contains the ID and type of the document. If you do not specify an ID or type when you create a document, the system automatically generates one for the document.
- If you enable the Auto Indexing feature and the specified index name does not exist, the system automatically creates an index when creating the document. The Auto Indexing feature is disabled by default. For information about how to enable this feature, see [Configure a YML file](#).

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "my_type",
  "_id": "AV4Jlvi15ny3i8DCdK1H",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "created": true
}
```

10.4.2. Update a document

This topic describes how to use Elasticsearch to update a document.

[Access an Elasticsearch cluster](#) and send a POST request to update a document in the cluster.

```
curl -XPOST http://<HOST>:<PORT>/my_index/my_type/<doc_id>
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200`.
- `my_index` : the name of the index.
- `my_type` : the type of the document.
- `<doc_id>` : the ID of the document.

Example:

```
curl -XPOST http://<HOST>:<PORT>/my_index/my_type/AV4Jlvi15ny3i8DCdK1H
-d '{"title": "FOUR UPDATED", "TAGS": ["RUBY", "PHP"]}'
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "my_type",
  "_id": "AV4Jlvi15ny3i8DCdK1H",
  "_version": 2,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 1,
    "failed": 0
  },
  "created": false
}
```

10.4.3. Retrieve a document

This topic describes how to use Elasticsearch to retrieve a document.

Access an Elasticsearch cluster and send a GET request to retrieve a document in the cluster.

Example:

```
curl -XGET http://<HOST>:<PORT>/my_index/my_type/AV4JIVI15NY3I8DCDK1H
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200`.
- `my_index` : the name of the index.
- `my_type` : the type of the document.
- `AV4JIVI15NY3I8DCDK1H` : the ID of the document.

If the command is successfully executed, the following result is returned:

```
{
  "_INDEX": "MY_INDEX",
  "_TYPE": "MY_TYPE",
  "_ID": "AV4JIVI15NY3I8DCDK1H",
  "_VERSION": 2,
  "EXISTS": TRUE,
  "_SOURCE": {
    "TITLE": "FOUR UPDATED", "TAGS": ["RUBY", "PHP"]
  }
}
```

10.4.4. Search for documents

This topic describes how to use Elasticsearch to search for documents.

Access an Elasticsearch cluster and send a GET or POST request to search for documents in the cluster. You can specify URI parameters in the request. Examples:

```
curl -XGET http://<HOST>:<PORT>/_search
curl -XGET http://<HOST>:<PORT>/<index_name>/_search
curl -XGET http://<HOST>:<PORT>/<index_name>/<type_name>/_search
```

To search for documents in which the title field contains the T keyword, run the following command:

```
curl -XGET http://<HOST>:<PORT>/my_index/my_type/_search?q=title:T*
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200`.
- `my_index` : the name of the index.
- `my_type` : the type of the document.

10.4.5. Perform a complex search

This topic describes how to use Elasticsearch to perform a complex search.

Access an Elasticsearch cluster and send a POST request to perform a complex search on documents in the cluster. Example:

```
$ curl -XPOST http://<HOST>:<PORT>/my_index/my_type/_search?pretty=true -d '{
  "query": {
    "query_string": {"query": "*" }
  },
  "facets": {
    "tags": {
      "terms": {"field": "tags"}
    }
  }
}'
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200` .
- `my_index` : the name of the index.
- `my_type` : the type of the document.

 **Note** The `?pretty=true` parameter is used to make the result easier to understand.

10.4.6. Delete a document

This topic describes how to use Elasticsearch to delete a document.

Access an [Elasticsearch cluster](#) and call the DELETE operation to delete a document in the cluster.

Delete a document with its ID specified

```
curl -XDELETE http://<HOST>:<PORT>/my_index/my_type/{ID}
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200` .
- `my_index` : the name of the index.
- `my_type` : the type of the document.
- `ID` : the ID of the document.

Delete a specified type of documents

```
curl -XDELETE http://<HOST>:<PORT>/my_index/my_type/
```

- `<HOST>` : the internal network endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200` .
- `my_index` : the name of the index.
- `my_type` : the type of the documents.

Delete documents in a specified index

```
curl -XDELETE http://<HOST>:<PORT>/{my_index}
```

- `<HOST>` : the internal endpoint of the Elasticsearch cluster.
- `<PORT>` : the internal port of the Elasticsearch cluster. The default port is `9200` .

- `my_index` : the name of the index.

10.5. Elasticsearch test

After you create an Elasticsearch instance, you can log on to the Kibana console integrated into the Elasticsearch console and test the search function on the Dev Tools page. You can also run the curl command in an ECS instance that meets the requirements to perform the test.

10.5.1. Use a curl command to access an Elasticsearch cluster over port 9200

This topic describes how to use a curl command to access an Elasticsearch cluster over port 9200.

The following example shows how to access an Elasticsearch cluster, with its username and password specified:

```
curl -u username:password 'http://<HOST>:9200/filebeat/my_type/?pretty -d '{"title": "One", "tags": ["ruby"]}'
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "filebeat",
  "_type": "my_type",
  "_id": "AV-bTkaTwdiHxfaSqlAt",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "created": true
}
```

The following example shows how to access an Elasticsearch cluster, with its username and password not specified:

```
curl http://<HOST>:9200/index_name/type_name -XPOST -d '{"title": "One", "tags": ["ruby"]}'
```

10.5.2. Use Python to access an Elasticsearch cluster over port 9200

This topic describes how to use Python to access an Elasticsearch cluster over port 9200.

```
from elasticsearch import Elasticsearch, RequestsHttpConnection
import certifi
es = Elasticsearch(
    ['<HOST>'],
    http_auth=('username', 'password'),
    port=9200,
    use_ssl=False
)
res = es.index(index="my_index", doc_type="my_type", id=1, body={"title": "One", "tags": ["ruby"]})
print(res['_created'])
res = es.get(index="my-index", doc_type="my-type", id=1)
print(res['_source'])
```

10.5.3. Use Java REST Client to access an Elasticsearch cluster over port 9200

This topic describes how to use Java REST Client to access an Elasticsearch cluster over port 9200 and provides related precautions.

Precautions

- The open source Elasticsearch team no longer maintains Transport Client. We recommend that you do not use Transport Client to access an Elasticsearch cluster. If you use Transport Client 5.5.3 to access an Elasticsearch cluster, the system displays the "NoNodeAvailableException" error message. We recommend that you use [Java Low Level REST Client](#) to access an Elasticsearch cluster.
- Java REST Client described in this topic is applicable only to Elasticsearch V5.5.3 clusters. For more information about how to use Java REST Client to access an Elasticsearch V6.3.2 cluster, see [Java REST Client 6.3.2](#).
- The version of Java REST Client must be the same as that of your Elasticsearch cluster.

Prerequisites

- An Elasticsearch cluster is created, and the Auto Indexing feature is enabled for the cluster.
For more information, see [Create an Elasticsearch cluster](#) and [Configure a YML file](#).
- The relevant JDK is installed, and environment variables are configured. The JDK version must be 1.8 or later.

Sample code

```
import org.apache.http.HttpEntity;
import org.apache.http.HttpHost;
import org.apache.http.auth.AuthScope;
import org.apache.http.auth.UsernamePasswordCredentials;
import org.apache.http.client.CredentialsProvider;
import org.apache.http.entity.ContentType;
import org.apache.http.impl.client.BasicCredentialsProvider;
import org.apache.http.impl.nio.client.HttpAsyncClientBuilder;
import org.apache.http.nio.entity.NStringEntity;
import org.apache.http.util.EntityUtils;
import org.elasticsearch.client.Response;
import org.elasticsearch.client.RestClient;
import org.elasticsearch.client.RestClientBuilder;
import java.io.IOException;
import java.util.Collections;
public class RestClientTest {
    public static void main(String[] args){
        final CredentialsProvider credentialsProvider = new BasicCredentialsProvider();
        credentialsProvider.setCredentials(AuthScope.ANY,
            new UsernamePasswordCredentials("<USER NAME>", "<PASSWORD>"));
        RestClient restClient = RestClient.builder(new HttpHost("<HOST>", 9200))
            .setHttpClientConfigCallback(new RestClientBuilder.HttpClientConfigCallback() {
                @Override
                public HttpAsyncClientBuilder customizeHttpClient(HttpAsyncClientBuilder httpClientBuilder) {
                    return httpClientBuilder.setDefaultCredentialsProvider(credentialsProvider);
                }
            }).build();
        try {
            //index a document
            HttpEntity entity = new NStringEntity("{\"user\":\"kimchy\"}", ContentType.APPLICATION_JSON);
            Response indexResponse = restClient.performRequest(
                "PUT",
                "/index/type/123",
                Collections.<String, String>emptyMap(),
                entity);
            //search a document
            Response response = restClient.performRequest("GET", "/index/type/123",
                Collections.singletonMap("pretty", "true"));
            System.out.println(EntityUtils.toString(response.getEntity()));
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

- <USER NAME> : the username of your Elasticsearch cluster.
- <PASSWORD> : the password of your Elasticsearch cluster.
- <HOST> : the internal endpoint of your Elasticsearch cluster. You can obtain the endpoint from the Basic Information page of the cluster.

11. Elasticsearch (on k8s)

11.1. What is Apsara Stack Elasticsearch?

Open source Elasticsearch is a Lucene-based, distributed, real-time search and analytics engine. It is a product released under the Apache License. Elasticsearch is a popular search engine for enterprises. It provides distributed services, allowing you to store, query, and analyze large amounts of datasets in near real time. Elasticsearch is typically used as a basic engine or technology to support complex queries and high-performance applications.

Apsara Stack Elasticsearch provides fully-managed Elasticsearch services. It supports multiple versions of open source Elasticsearch and is compatible with all open source Elasticsearch features. Apsara Stack Elasticsearch offers an optimized kernel and provides the multi-tenancy, high availability, and auto scaling features. In addition to the features of open source Elasticsearch, Apsara Stack Elasticsearch allows you to create a cluster in a visualized manner, use Migration Assistant to migrate data, manage repositories, create snapshots, manage plug-ins, and perform cluster O&M.

Each Apsara Stack Elasticsearch cluster is composed of multiple nodes. The management nodes of Apsara Stack Elasticsearch support high availability. Management node failures do not affect the running of Apsara Stack Elasticsearch. In addition, Apsara Stack Elasticsearch can store audit logs and automatically dump them to the specified directory on a specific node for long-term storage and management.

11.2. Quick start

11.2.1. Log on to the Elasticsearch console

This topic describes how to log on to the Elasticsearch console.

Prerequisites

- The URL of the Apsara Uni-manager Management Console is obtained from the deployment personnel before you log on to the Apsara Uni-manager Management Console.
- A browser is available. We recommend that you use the Google Chrome browser.

Procedure

1. In the address bar, enter the URL of the Apsara Uni-manager Management Console. Press the Enter key.
2. Enter your username and password.

Obtain the username and password that you can use to log on to the console from the operations administrator.

 **Note** When you log on to the Apsara Uni-manager Management Console for the first time, you must change the password of your username. Your password must meet complexity requirements. The password must be 8 to 20 characters in length and must contain at least two of the following character types:

- Uppercase or lowercase letters
- Digits
- Special characters, which include ! @ # \$ %

3. Click **Login**.
4. In the top navigation bar, choose **Products > Big Data > Elasticsearch(k8s)**.

5. Specify **Organization** and **Region** and click **Elasticsearch on k8s**.

11.2.2. Access an Elasticsearch cluster

You can access an Elasticsearch cluster from an Elastic Compute Service (ECS) instance or the Kibana console.

Prerequisites

- Operations and maintenance (O&M) personnel have created an Elasticsearch cluster under your organization.
For more information, see [Elasticsearch Operations and Maintenance Guide](#).
- An ECS instance is created. The ECS instance must reside in the same Virtual Private Cloud (VPC), Kubernetes cluster, region, and zone as the Elasticsearch cluster.

Use an ECS instance to access an Elasticsearch cluster

1. Log on to the ECS instance over SSH and install the cURL tool.

 **Note** For more information about other logon methods, see [ECS User Guide](#).

2. Run the following command to connect to the internal endpoint of the Elasticsearch cluster.

```
curl http://<HOST>:<PORT>
```

- **<HOST>** : the internal endpoint of the Elasticsearch cluster. You can obtain the information from the [Basic Information](#) page of the cluster.

 **Note** To go to the [Basic Information](#) page, [log on to the Elasticsearch console](#), find the cluster, and click its ID in the [Instance ID/Name](#) column.

- **<PORT>** : the internal port of the Elasticsearch cluster. You can obtain the information from the [Basic Information](#) page. The default port is 9200.

Use the Kibana console to access an Elasticsearch cluster

1. [Log on to the Elasticsearch console](#).
2. Find the target cluster and click its ID in the [Instance ID/Name](#) column.
3. In the left-side navigation pane, click **Data Visualization**.
4. Click **Console** in the Kibana section.
5. On the page that appears, enter the username and password and click **Log in**. The username is elastic. You can obtain the password from O&M personnel.
6. In the left-side navigation pane, click the **Dev Tools** icon. On the **Console** tab, run the command shown in the following figure.

The screenshot shows the Kibana Dev Tools interface. On the left, the console displays a REST client request:

```

1 GET _search
2 {
3   "query": {
4     "match_all": {}
5   }
6 }

```

On the right, the response is shown as a JSON object:

```

1 {
2   "took": 128,
3   "timed_out": false,
4   "_shards": {
5     "total": 24,
6     "successful": 24,
7     "skipped": 0,
8     "failed": 0
9   },
10  "hits": {
11    "total": {
12      "value": 10000,
13      "relation": "gte"
14    },
15    "max_score": 1.0,
16    "hits": [
17      {
18        "_index": ".kibana_1",
19        "_type": "_doc",
20        "_id": "space:default",
21        "_score": 1.0,
22        "_source": {
23          "space": {
24            "name": "Default",
25            "description": "This is your default space!",
26            "color": "#00bfb3",
27            "disabledFeatures": [ ],
28            "_reserved": true
29          },
30          "type": "space",
31          "migrationVersion": {
32            "space": "6.6.0"
33          },
34          "updated_at": "2020-07-30T07:35:49.734Z"
35        }

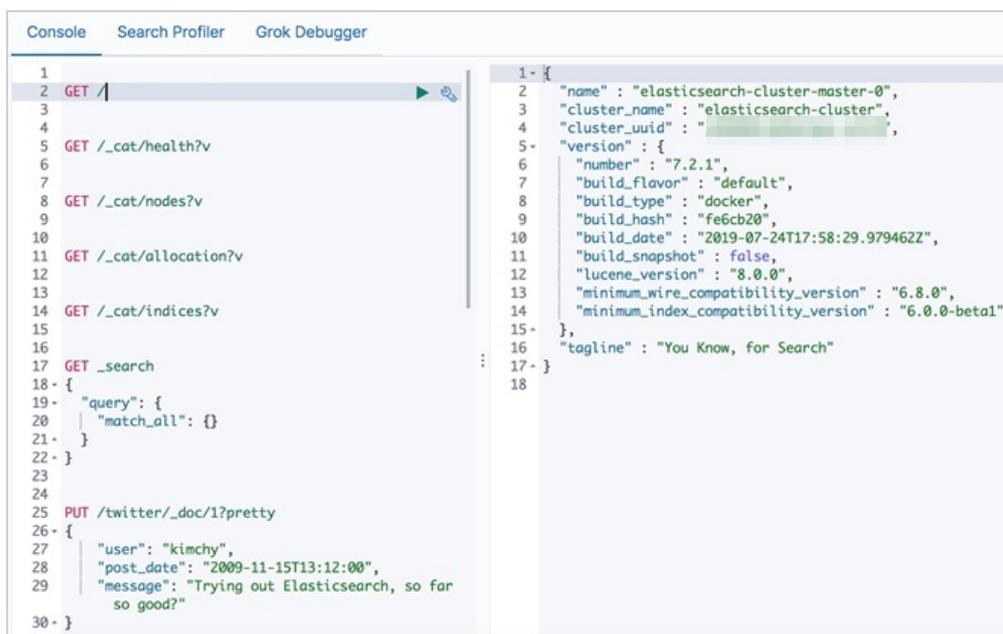
```

11.2.3. View the information of an Elasticsearch cluster

This topic describes how to view the information of an Elasticsearch cluster from the Kibana console.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following commands to view the basic information of the cluster.



- o View the version of the cluster

```
GET /
```

- o View the health status of the cluster

```
GET /_cat/health?v
```

- o View the information of nodes in the cluster

```
GET /_cat/nodes?v
```

- o View JVM heap memory

```
GET /_nodes/stats/jvm?pretty
```

- o View information of disks

```
GET /_cat/allocation?v
```

- o View information of indexes

```
GET /_cat/indices?v
```

11.2.4. Create an index

Before you use Elasticsearch to process documents, you must create an index. This topic describes how to create an index.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to create an index:

```
PUT my_index?include_type_name=true
{
  "settings": {
    "index": {
      "number_of_shards": "5",
      "number_of_replicas": "1"
    }
  },
  "mappings": {
    "_doc": {
      "properties": {
        "post_date": {
          "type": "date"
        },
        "tags": {
          "type": "keyword"
        },
        "title": {
          "type": "text",
          "analyzer": "cjk"
        }
      }
    }
  }
}
```

Notice

- The preceding command is only for your reference. You can customize the parameters as needed. For more information, see [open source Elasticsearch documentation](#).
- You can configure the YAML configuration file of your Elasticsearch cluster to enable the Auto Indexing feature. After this feature is enabled, when you create a document, the system automatically creates an index for the document. Auto Indexing is used only for testing purposes. We recommend that you do not use this feature in production.
- Mapping types are deprecated in Elasticsearch V7.0.0 and later. Versions earlier than Elasticsearch V7.0.0 still support mapping types. For more information, see [open source Elasticsearch documentation](#).

In the preceding example, an index named `my_index` is created, and the type of the index is `_doc`. The index is split into five shards, has one replica for each shard, and uses the `cjk` analyzer.

If the command is successfully executed, the following result is returned:

```
{
  "acknowledged": true,
  "shards_acknowledged": true,
  "index": "my_index"
}
```

11.2.5. Manage documents

11.2.5.1. Create a document

This topic describes how to use Elasticsearch to create a document.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to create a document:

```
curl -XPUT http://<HOST>:<PORT>/<index_name>/_doc/<ID>
{
  "<field_name>": "<value>"
}'
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is 9200.
- o `<index_name>` : the name of the index.
- o `_doc` : the type of the index. The index type of Elasticsearch V7.0 and later must be `_doc`.
- o `<ID>` : the ID of the document.
- o `<field_name>` : the name of the field.
- o `value` : the value of the field.

Examples:

```
curl -XPUT http://localhost:9200/my_index/_doc/1?pretty' -d '
{
  "title": "One",
  "tags": ["ruby"],
  "post_date": "2009-11-15T13:00:00"
}'
```

```
curl -XPUT http://localhost:9200/my_index/_doc/2?pretty' -d '
{
  "title": "Two",
  "tags": ["ruby"],
  "post_date": "2009-11-15T14:00:00"
}'
```

If the commands are successfully executed, the following results are returned:

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "1",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "_seq_no": 0,
  "_primary_term": 1
}
```

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "2",
  "_version": 1,
  "result": "created",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "_seq_no": 1,
  "_primary_term": 1
}
```

For more information, see [open source Elasticsearch documentation](#).

11.2.5.2. Update a document

This topic describes how to use Elasticsearch to update a document.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run one of the following commands to update a document:

Update a whole document

```
curl -XPUT http://<HOST>:<PORT>/<index_name>/_doc/<ID>
{
  "<field_name>": "<value>"
}'
```

Make partial updates to a document

```
curl -XPOST http://<HOST>:<PORT>/<index_name>/_update/<ID>
{
  "script": {
    "source": "ctx._source.<field_name> = '<value>'"
  }
}'
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is `9200`.
- o `<index_name>` : the name of the index.
- o `_doc` : the type of the index.
- o `<ID>` : the ID of the document.
- o `<field_name>` : the name of the field.
- o `value` : the value of the field.

Examples:

- o Update a whole document

```
curl -XPUT http://localhost:9200/my_index/_doc/1?pretty' -d '{
  "title": "One World One Dream",
  "tags": ["ruby"],
  "post_date": "2009-11-15T13:00:00"
}'
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "1",
  "_version": 2,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "_seq_no": 1,
  "_primary_term": 1
}
```

- o Make partial updates to a document

```
curl -XPOST http://localhost:9200/my_index/_update/1?pretty' -d '{
  "script": {
    "source": "ctx._source.title='One World'"
  }
}'
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "1",
  "_version": 5,
  "result": "updated",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "_seq_no": 4,
  "_primary_term": 1
}
```

You can also call the update operation to update documents. For more information, see [open source Elasticsearch documentation](#).

11.2.5.3. Retrieve a document

This topic describes how to use Elasticsearch to retrieve a document.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to retrieve a document:

```
curl -XGET http://<HOST>:<PORT>/<index_name>/_doc/<ID>
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster.
- o `<index_name>` : the name of the index.
- o `_doc` : the type of the index.
- o `<ID>` : the ID of the document.

Example:

```
curl -XGET http://localhost:9200/my_index/_doc/1
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "1",
  "_version": 5,
  "_seq_no": 4,
  "_primary_term": 1,
  "found": true,
  "_source": {
    "title": "One World",
    "tags": [
      "ruby",
      "blue"
    ],
    "post_date": "2009-11-15T13:00:00"
  }
}
```

For more information, see [open source Elasticsearch documentation](#).

11.2.5.4. Search for documents

This topic describes how to use Elasticsearch to search for documents.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to search for documents. URI parameters in the command are used to specify the object you want to search.

```
curl -XGET http://<HOST>:<PORT>/_search
curl -XGET http://<HOST>:<PORT>/<index_name>/_search
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is 9200.

- o `<index_name>` : the name of the index.

To search for documents in which the title field contains the T keyword, run the following command:

```
curl -XGET http://localhost:9200/my_index/_search?q=title:T*
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 689,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 1,
      "relation": "eq"
    },
    "max_score": 1.0,
    "hits": [
      {
        "_index": "my_index",
        "_type": "_doc",
        "_id": "2",
        "_score": 1.0,
        "_source": {
          "title": "Two",
          "tags": [
            "ruby"
          ]
        },
        "post_date": "2009-11-15T14:00:00"
      }
    ]
  }
}
```

For more information, see [open source Elasticsearch documentation](#).

11.2.5.5. Perform a complex search

This topic describes how to use Elasticsearch to perform a complex search.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to perform a complex search:

```
curl -XGET http://<HOST>:<PORT>/<index_name>/_search?pretty=true -d '{
  "query":{
    ""
  }
}'
```

- o **<HOST>** : the endpoint of the Elasticsearch cluster.
- o **<PORT>** : the port of the Elasticsearch cluster. The default port is 9200.
- o **<index_name>** : the name of the index.
- o **query** : the request body.

 **Note** `?pretty=true` is used to make the returned result easier to understand.

Examples:

- o Search based on a time range

```
curl -XGET http://localhost:9200/my_index/_search?pretty=true -d '{
  "query": {
    "range": {
      "post_date": { "from": "2009-11-15T13:00:00", "to": "2009-11-15T14:00:00" }
    }
  }
}'
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 2,
      "relation": "eq"
    },
    "max_score": 1.0,
    "hits": [
      {
        "_index": "my_index",
        "_type": "_doc",
        "_id": "2",
        "_score": 1.0,
        "_source": {
          "title": "Two",
          "tags": [
            "ruby"
          ],
          "post_date": "2009-11-15T14:00:00"
        }
      },
      {
        "_index": "my_index",
        "_type": "_doc",
        "_id": "1",
        "_score": 1.0,
        "_source": {
          "title": "One World",
          "tags": [
            "ruby"
          ],
          "post_date": "2009-11-15T13:00:00"
        }
      }
    ]
  }
}
```

- Search based on a keyword

```
curl -XGET "http://localhost:9200/my_index/_search?pretty=true" -H 'Content-Type: application/json' -d '{
  "query": {
    "simple_query_string": {
      "query": "One +(python | ruby) -Two",
      "fields": ["title^5", "tags"],
      "default_operator": "or"
    }
  }
}'
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 1,
      "relation": "eq"
    },
    "max_score": 2.2378674,
    "hits": [
      {
        "_index": "my_index",
        "_type": "_doc",
        "_id": "1",
        "_score": 2.2378674,
        "_source": {
          "title": "One World",
          "tags": [
            "ruby"
          ],
          "post_date": "2009-11-15T13:00:00"
        }
      }
    ]
  }
}
```

For more information, see [open source Elasticsearch documentation](#).

11.2.5.6. Perform statistical analytics

This topic describes how to use Elasticsearch to perform statistical analytics on documents.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to perform statistical analytics on documents:

```
curl -XPOST http://<HOST>:<PORT>/<index_name>/_search?pretty=true -d '{
  "size": 0,
  "aggs": {
    "group_by_tags": {
      "terms": {
        "field": "tags"
      }
    }
  }
}'
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is 9200.
- o `<index_name>` : the name of the index.

 **Note** `?pretty=true` is used to make the returned result easier to understand.

Example:

```
curl -XPOST http://localhost:9200/my_index/_search?pretty=true -d '{
  "size": 0,
  "aggs": {
    "group_by_tags": {
      "terms": {
        "field": "tags"
      }
    }
  }
}'
```

If the command is successfully executed, the following result is returned:

```
{
  "took": 1,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": {
      "value": 2,
      "relation": "eq"
    },
    "max_score": null,
    "hits": []
  },
  "aggregations": {
    "group_by_tags": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [
        {
          "key": "ruby",
          "doc_count": 2
        }
      ]
    }
  }
}
```

11.2.5.7. Delete a document

This topic describes how to use Elasticsearch to delete a document.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to delete a document:

```
curl -XDELETE http://<HOST>:<PORT>/<index_name>/_doc/<ID>
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is 9200.
- o `<index_name>` : the name of the index.
- o `_doc` : the type of the index.
- o `<ID>` : the ID of the document.

Example:

```
curl -XDELETE http://localhost:9200/my_index/_doc/2
```

If the command is successfully executed, the following result is returned:

```
{
  "_index": "my_index",
  "_type": "_doc",
  "_id": "2",
  "_version": 2,
  "result": "deleted",
  "_shards": {
    "total": 2,
    "successful": 2,
    "failed": 0
  },
  "_seq_no": 4,
  "_primary_term": 1
}
```

For more information, see [open source Elasticsearch documentation](#).

11.2.6. Delete an index

This topic describes how to call the delete operation for Elasticsearch to delete an index.

Procedure

1. Access your Elasticsearch cluster from the Kibana console. For more information, see [Access an Elasticsearch cluster](#).
2. Run the following command to delete an index:

```
curl -XDELETE http://<HOST>:<PORT>/<index_name>
```

- o `<HOST>` : the endpoint of the Elasticsearch cluster.
- o `<PORT>` : the port of the Elasticsearch cluster. The default port is 9200.
- o `<index_name>` : the name of the index.

Example:

```
curl -XDELETE http://localhost:9200/my_index
```

If the command is successfully executed, the following result is returned:

```
{
  "acknowledged": true
}
```

For more information, see [open source Elasticsearch documentation](#).

11.3. Manage clusters

11.3.1. Customize a cluster list

You can customize columns that are displayed in a cluster list. This makes it easy for you to view cluster information. This topic describes how to customize the columns.

Procedure

1. [Log on to the Elasticsearch console](#).

2. In the upper-right corner of the **Elasticsearch Instances** page, click the  icon.
3. In the **Select Filters** dialog box, select columns as needed.
4. Click **OK**.
The system then displays the cluster list based on the selected items.

11.3.2. Export a cluster list

You can export the list of Elasticsearch clusters based on selected columns. This topic describes how to export a cluster list.

Procedure

1. [Log on to the Elasticsearch console](#).
2. In the upper-right corner of the **Elasticsearch Instances** page, click the  icon.
3. In the **Export** dialog box, specify **Export Mode** and **Translate Heading**, and select the columns that you want to export.
 - **Export Mode**: the mode used to export the cluster list. Valid values: All and Selected. Default value: All.
 - **Translate Heading**: specifies whether to translate headings. Valid values: Yes and No. Default value: Yes. If you set this parameter to Yes, exported headings are in Chinese. If you set this parameter to No, exported headings are in English, such as Instance ID.
 - **Custom**: Select columns that you want to export.
4. Click **OK**.

11.3.3. Refresh the information of a cluster

If the information of an Elasticsearch cluster, such as its status, is not refreshed in time in the console, you can manually refresh the page to obtain the up-to-date information. This topic describes how to refresh the information of an Elasticsearch cluster.

Procedure

1. [Log on to the Elasticsearch console](#).
2. Use one of the following methods to refresh the information of a cluster:
 - On the **Elasticsearch Instances** page, click the  icon in the upper-right corner.
 - On the **Elasticsearch Instances** page, find the target cluster and click its ID in the **Instance ID/Name** column. On the **Basic Information** page, click **Refresh** in the upper-right corner.

11.3.4. View the basic information of an Elasticsearch cluster

On the **Basic Information** page of an Elasticsearch cluster, you can view the information of the cluster, such as the ID, name, internal endpoint, and status. This topic describes how to view the basic information of a cluster.

Procedure

1. [Log on to the Elasticsearch console.](#)
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. On the **Basic Information** page, view the basic information of the cluster.

Section	Parameter	Description
Basic Information section	Instance ID	The unique ID of the cluster.
	Name	The name of the cluster. By default, the name of a cluster is the same as its ID. Cluster names are configurable. You can search for clusters by name.
	Version	Valid values: 6.3.2, 6.7.0, 6.8.2, 7.2.1, and 7.4.0.
	Region	The region where the cluster resides.
	Zone	The zone where the cluster resides.
	Internal Network Address	The internal endpoint of the cluster. In a Virtual Private Cloud (VPC), you can use the internal endpoint to access the cluster.
Instance Statistics	Internal Network Port	Elasticsearch supports the following ports: <ul style="list-style-type: none"> ◦ Port 9200 for HTTP ◦ Port 9300 for TCP
	Status	The status of the cluster. A cluster has the following states: Active (green), Initializing (yellow), Unhealthy (red), Paused (red), and Expired (gray).
Configuration Info	Created At	The time when the cluster was created.
	None	The configuration of the cluster. You must specify the configuration when you create the cluster. Related configuration items include Data Nodes, Dedicated Master Nodes (optional), Client Nodes (optional), and Kibana Nodes. For more information, see Elasticsearch Operations and Maintenance Guide.

11.3.5. Log on to the Kibana console

Apsara Stack Elasticsearch provides the Kibana console for you to scale your businesses. The Kibana console is a part of the Elastic ecosystem and is seamlessly integrated into Elasticsearch. This allows you to view the status of your Elasticsearch cluster in real time and manage the cluster. This topic describes how to log on to the Kibana console.

Procedure

1. [Log on to the Elasticsearch console.](#)
2. Find the target cluster and click its ID in the **Instance ID/Name** column.
3. In the left-side navigation pane, click **Data Visualization**.
4. In the **Kibana** section of the page that appears, click **Console**.
5. On the page that appears, enter the username and password and click **Log in**. The username is elastic. The

password is the one that is specified when the cluster is created. You can obtain the password from operations and maintenance (O&M) personnel.

11.3.6. Create a snapshot and restore data

You can call the snapshot operation to back up or restore data for your Apsara Stack Elasticsearch cluster. The snapshot operation retrieves the status and data of your cluster and stores them to a shared repository.

The first snapshot is a full copy of the data in a cluster. Subsequent snapshots store only incremental data. When you create a subsequent snapshot, the system only adds data to or removes data from the previous snapshot. This means that it requires less time to create a subsequent snapshot than the first snapshot.

Precautions

- This topic uses the following markers to provide descriptions for code: <1>, <2>, and <3>. Remove these markers before you run the code.
- You can run all the commands provided in this topic in the Kibana console of your Elasticsearch cluster. For more information, see [Log on to the Kibana console](#).

Create a repository

```
PUT _snapshot/my_backup
{
  "type": "oss",
  "settings": {
    "endpoint": "http://oss-cn-hangzhou-internal.aliyuncs.com", <1>
    "access_key_id": "xxxx",
    "secret_access_key": "xxxxxx",
    "bucket": "xxxxxx", <2>
    "compress": true,
    "base_path": "snapshot/" <3>
  }
}
```

- <1>: The `endpoint` parameter specifies the internal endpoint of an Object Storage Service (OSS) bucket.
- <2>: The `bucket` parameter specifies the name of the OSS bucket.
- <3>: The `base_path` parameter specifies the path of the repository. The default value is the root directory.

Set the size of each part

You can upload large volumes of data to an OSS bucket in multiple parts. When you upload the data, you can use the `chunk_size` parameter to set the size of each part. Example:

```
POST _snapshot/my_backup/ <1>
{
  "type": "oss",
  "settings": {
    "endpoint": "http://oss-cn-hangzhou-internal.aliyuncs.com",
    "access_key_id": "xxxx",
    "secret_access_key": "xxxxxx",
    "bucket": "xxxxxx",
    "chunk_size": "500mb",
    "base_path": "snapshot/" <2>
  }
}
```

- <1>: Use the POST method instead of the PUT method. The POST method updates repository settings.
- <2>: The `base_path` parameter specifies the path of the repository. The default value is the root directory.

Query repository information

```
GET _snapshot
```

You can also use the `GET _snapshot/my_backup` command to query the information of a specified repository.

Create a snapshot for all open indexes

The following command is a basic command that is used to create a snapshot:

```
PUT _snapshot/my_backup/snapshot_1
```

This command creates a snapshot named `snapshot_1` for all open indexes and stores the snapshot in a repository named `my_backup`. After you run the command, the system immediately returns a response and creates the snapshot at the backend.

If you want the system to return a response after the snapshot is created, specify the `wait_for_completion` parameter in the command. Example:

```
PUT _snapshot/my_backup/snapshot_1?wait_for_completion=true
```

After you run the command, the system does not return a response until the snapshot is created. If the size of an index is large, the response is returned after a longer period of time.

Create a snapshot for specified indexes

By default, a snapshot contains all open indexes. If Kibana is used when you create a snapshot, you may want to ignore all diagnostic indexes (the `.kibana` indexes) because of limited disk space. To create a snapshot for specified indexes, run the following command.

 **Notice** A repository stores multiple snapshots. Each snapshot is a copy of all indexes, specified indexes, or a single index in a cluster. When you create a snapshot, make sure that the snapshot name is unique.

```
PUT _snapshot/my_backup/snapshot_2
{
  "indices": "index_1,index_2"
}
```

The preceding command creates a snapshot only for the `index1` and `index2` indexes.

Query snapshot information

In some cases, you may need to query snapshot information. For example, a snapshot name that contains a date is hard to remember, such as `backup_2014_10_28`.

To query the information of a snapshot, send a `GET` request that contains both the repository name and snapshot name. Example:

```
GET _snapshot/my_backup/snapshot_2
```

The response contains detailed information of the snapshot:

```
{
  "snapshots": [
    {
      "snapshot": "snapshot_2",
      "indices": [
        ".marvel_2014_28_10",
        "index1",
        "index2"
      ],
      "state": "SUCCESS",
      "start_time": "2014-09-02T13:01:43.115Z",
      "start_time_in_millis": 1409662903115,
      "end_time": "2014-09-02T13:01:43.439Z",
      "end_time_in_millis": 1409662903439,
      "duration_in_millis": 324,
      "failures": [],
      "shards": {
        "total": 10,
        "failed": 0,
        "successful": 10
      }
    }
  ]
}
```

You can replace the snapshot name in the preceding command with `_all` to query all snapshots in the repository. Example:

```
GET _snapshot/my_backup/_all
```

Delete a snapshot

You can specify a repository name and a snapshot name in a `DELETE` request to delete the specified snapshot. Example:

```
DELETE _snapshot/my_backup/snapshot_2
```

Notice

- You can delete snapshots only by calling the `DELETE` operation. A snapshot is associated with other backup files. Some of the files may also be used by other snapshots. The `DELETE` operation does not delete files that are still being used by other snapshots. It deletes only the files that are associated with deleted snapshots and are no longer used by other snapshots.
- If you choose to manually delete a snapshot, you may delete files that are used by other snapshots. This can cause data loss.

Monitor snapshot creation progress

The `wait_for_completion` parameter provides a simple method for you to monitor the progress of a snapshot creation task. However, this parameter is not suitable for snapshot creation tasks of medium-size Elasticsearch clusters. You can use one of the following methods to query detailed information about a snapshot:

- Send a `GET` request with the snapshot name specified. Example:

```
GET _snapshot/my_backup/snapshot_3
```

If the system is creating the snapshot when you run the preceding command, the information of the creation task is returned, such as the start time and duration of the task.

 **Notice** The preceding command shares a thread pool with the command used to create a snapshot. Therefore, if you create a snapshot for large shards, the preceding command has to wait until the resources that are used by the snapshot creation command in the thread pool are released.

- Call the status operation to query the snapshot status.

```

{
  "snapshots": [
    {
      "snapshot": "snapshot_3",
      "repository": "my_backup",
      "state": "IN_PROGRESS", <1>
      "shards_stats": {
        "initializing": 0,
        "started": 1, <2>
        "finalizing": 0,
        "done": 4,
        "failed": 0,
        "total": 5
      },
      "stats": {
        "number_of_files": 5,
        "processed_files": 5,
        "total_size_in_bytes": 1792,
        "processed_size_in_bytes": 1792,
        "start_time_in_millis": 1409663054859,
        "time_in_millis": 64
      },
      "indices": {
        "index_3": {
          "shards_stats": {
            "initializing": 0,
            "started": 0,
            "finalizing": 0,
            "done": 5,
            "failed": 0,
            "total": 5
          },
          "stats": {
            "number_of_files": 5,
            "processed_files": 5,
            "total_size_in_bytes": 1792,
            "processed_size_in_bytes": 1792,
            "start_time_in_millis": 1409663054859,
            "time_in_millis": 64
          },
          "shards": {
            "0": {
              "stage": "DONE",
              "stats": {
                "number_of_files": 1,
                "processed_files": 1,
                "total_size_in_bytes": 514,
                "processed_size_in_bytes": 514,
                "start_time_in_millis": 1409663054862,
                "time_in_millis": 22
              }
            }
          }
        }
      },
      ...
    }
  ]
}

```

- <1>: the status of the snapshot. If a snapshot is being created, the value of the field is `IN_PROGRESS` .

- `<2>`: the number of shards that are being transmitted. If the value 1 is returned, a shard is being transmitted to the snapshot, and the other four shards have been transmitted.

The value of the `shards_stats` parameter contains the status of the snapshot and the statistics about each index and shard. This parameter allows you to obtain the detailed information of the snapshot creation progress. A shard can be in one of the following states:

- `INITIALIZING` : The shard is verifying the status of the cluster to check whether the shard can be stored in a snapshot. In most cases, this process is fast.
- `STARTED` : Data is being transmitted to the repository.
- `FINALIZING` : Data is transmitted, and the shard is sending snapshot metadata.
- `DONE` : A snapshot is created for the shard.
- `FAILED` : An error occurred during the snapshot creation. The shard, index, or snapshot cannot be processed. You can view logs for more information.

Cancel a snapshot

To cancel a snapshot, run the following command when the snapshot is being created:

```
DELETE _snapshot/my_backup/snapshot_3
```

This command stops the snapshot creation process and deletes the snapshot that is being created from the repository.

Restore indexes from a snapshot

To restore indexes from a snapshot, run the command that is used in the "" section on the Elasticsearch cluster that stores these indexes. You can use one of the following methods to restore indexes from a snapshot:

- To restore indexes from a specified snapshot, append the `_restore` parameter to the snapshot name in the command to run. Example:

```
POST _snapshot/my_backup/snapshot_1/_restore
```

After you run this command, the system restores all indexes in the snapshot. For example, if the `snapshot_1` snapshot contains five indexes, all these indexes are restored to the related Elasticsearch cluster. You can also specify the indexes that you want to restore. For more information, see the "" section.

- Restore specified indexes and rename the indexes. If you only want to verify or process the data in indexes and do not want to overwrite the data, use this method to restore the indexes.

```
POST /_snapshot/my_backup/snapshot_1/_restore
{
  "indices": "index_1", <1>
  "rename_pattern": "index_(.+)", <2>
  "rename_replacement": "restored_index_$1" <3>
}
```

In this example, the `index_1` index is restored to your Elasticsearch cluster and renamed `restored_index_1`.

- `<1>`: The system restores only the `index_1` index from the snapshot.
 - `<2>`: The system searches for the index that is being restored and matches the index name with the provided index pattern.
 - `<3>`: The system renames the matched index.
- If you want the system to return a response after it restores the index, specify the `wait_for_completion` parameter in the command. Example:

```
POST _snapshot/my_backup/snapshot_1/_restore?wait_for_completion=true
```

After you call the restore operation, the system immediately returns a response and restores the index at the backend. If you want the system to return a response after the index is restored, specify the `wait_for_completion` parameter.

Monitor index restoration progress

 **Note** Restoring data from a repository applies the existing restoration mechanism in Elasticsearch. Restoring shards from a repository is the same as restoring data from a node.

You can call the recovery operation to monitor the progress of an index restoration task.

- Monitor the restoration of a specified index.

```
GET restored_index_3/_recovery
```

The recovery operation is a general-purpose operation that shows the status of the shards that are being transmitted to your cluster.

- Monitor the restoration of all indexes on the cluster. This may include shards that are irrelevant to the restoration process.

```
GET /_recovery/
```

Sample response:

```
{
  "restored_index_3": {
    "shards": [ {
      "id": 0,
      "type": "snapshot", <1>
      "stage": "index",
      "primary": true,
      "start_time": "2014-02-24T12:15:59.716",
      "stop_time": 0,
      "total_time_in_millis": 175576,
      "source": { <2>
        "repository": "my_backup",
        "snapshot": "snapshot_3",
        "index": "restored_index_3"
      },
      "target": {
        "id": "ryqJ5lO5S4-ISFbGntkEkg",
        "hostname": "my.fqdn",
        "ip": "10.0.**.**",
        "name": "my_es_node"
      },
      "index": {
        "files": {
          "total": 73,
          "reused": 0,
          "recovered": 69,
          "percent": "94.5%" <3>
        },
        "bytes": {
          "total": 79063092,
          "reused": 0,
          "recovered": 68891939,
          "percent": "87.1%"
        },
        "total_time_in_millis": 0
      },
      "translog": {
        "recovered": 0,
        "total_time_in_millis": 0
      },
      "start": {
        "check_index_time": 0,
        "total_time_in_millis": 0
      }
    }
  ]
}
```

- <1>: The `type` parameter indicates the type of restoration. The value `snapshot` indicates that the shard is being restored from a snapshot.
- <2>: The `source` parameter indicates the source snapshot and repository.
- <3>: The `percent` parameter indicates the progress of the restoration task. The value `94.5%` indicates that 94.5% of the shard files are restored.

The response lists all indexes that are being restored and the shards in these indexes. Each shard has statistics about the start or end time, duration, restoration progress, and bytes transmitted.

Cancel index restoration

To cancel index restoration, you only need to delete the indexes that are being restored. A restoration process is a shard restoration process. You can call the DELETE operation to modify the status of a cluster and cancel index restoration. Example:

```
DELETE /restored_index_3
```

If you run the preceding command when the `restored_index_3` index is being restored, the system stops the restoration and deletes the data that has been restored to your cluster. For more information, see [Snapshot And Restore](#).

Use a snapshot to migrate data

To use a snapshot to migrate data from an Elasticsearch cluster to another, perform the following steps:

1. Back up a snapshot to OSS.
2. Create a snapshot repository on the destination cluster. The repository must use the OSS bucket that stores the snapshot.
3. Set the `base_path` parameter to the path of the snapshot.
4. Run the data restoration command on the destination cluster.

 **Note** These steps provide a simple solution to snapshot-based data migration. For more information, see [Use a snapshot stored in OSS to migrate Elasticsearch data](#).

11.4. Use plug-ins

11.4.1. Use the a-pack-xdcr plug-in to replicate data across Elasticsearch clusters

This topic describes how to use the a-pack-xdcr plug-in for Elasticsearch to replicate data across Elasticsearch clusters in different data centers. This implements remote disaster recovery. Only Elasticsearch V6.7.0 and V6.8.2 clusters support this plug-in.

Prerequisites

- The a-pack-xdcr plug-in is installed on the source and destination Elasticsearch clusters.
You can contact O&M personnel to install the plug-in. For more information, see [Elasticsearch Operations and Maintenance Guide](#).
- The IP address of the source Elasticsearch cluster is obtained.
You can obtain the IP address and port number of the source Elasticsearch cluster by running the `kubectl get svc -n <elasticsearch namespace>` command on the relevant Kubernetes server. You can also obtain the information from O&M personnel.

Context

Elasticsearch provides the cross data center replication (XDCR) feature. It allows you to replicate data across Elasticsearch clusters to implement remote high availability (HA). Core link applications must be able to withstand the impact of service interruptions in data centers or regions. The native capabilities of XDCR can meet disaster recovery and HA requirements across data centers without requiring additional technologies. You can use the XDCR feature to replicate data to data centers that are located in close proximity to users or application servers. This reduces latency and costs. For example, you can replicate product lists or reference

datasets to 20 or more data centers around the world to minimize the distance between data and application servers.

Procedure

1. Log on to the Kibana console of the destination Elasticsearch cluster. For more information, see [Log on to the Kibana console](#).
2. In the left-side navigation pane, click **Dev Tools**.
3. On the **Console** tab, run the following command to create a remote repository in the destination Elasticsearch cluster and connect the repository to the source Elasticsearch cluster:

```
PUT _cluster/settings
{
  "persistent": {
    "cluster": {
      "remote": {
        "leader": {
          "seeds": [
            "<your cluster IP>:<your cluster port>"
          ]
        }
      }
    }
  }
}
```

 **Note** Replace `<your cluster IP>` with the IP address of the source cluster. Replace `<your cluster port>` with the port number of the source cluster. In most cases, the port number is 9300.

4. Run the following command to replicate data in an index:

```
PUT _xdr/leader/<your_index>
```

 **Note** Replace `<your_index>` with the name of the index whose data you want to replicate.

5. Run the following command to view the status of the replication task:

```
GET _cat/xdr?v
```

If the command is successfully executed, the following result is returned.

index	repository	shard	localSeqNo	remoteSeqNo
twitter	leader	4	1	1
twitter	leader	1	0	0
twitter	leader	2	7	7
twitter	leader	3	3	3
twitter	leader	0	-1	-1

Note `index` indicates the index name. `repository` indicates the destination cluster. `shard` indicates the index shard. `localSeqNo` indicates the maximum sequence number of the source cluster. `remoteSeqNo` indicates the maximum sequence number of the destination cluster.

6. Run the following command to view replicated data:

```
GET /<your_index>/_search
```

Note Replace `<your_index>` with the name of the index that stores the replicated data.

11.4.2. Use the `opendistro_sql` plug-in to query cluster data by executing SQL statements

This topic describes how to use the `opendistro_sql` plug-in for Elasticsearch to query data in an Elasticsearch cluster by executing SQL statements.

Prerequisites

The `opendistro_sql` plug-in is installed. You can contact O&M personnel to install the plug-in. For more information, see [Elasticsearch Operations and Maintenance Guide](#).

Procedure

1. Log on to the Kibana console of your Elasticsearch cluster. For more information, see [Log on to the Kibana console](#).
2. In the left-side navigation pane, click **Dev Tools**.
3. On the **Console** tab, run one of the following commands to query cluster data by executing SQL statements.

Elasticsearch supports the following SQL query syntax:

Note Replace `<your_index>` in the following code with the name of the index whose data you want to query.

- o Syntax for an SQL query in a GET request

```
GET _opendistro/_sql?sql=select * from <your_index> limit 50
```

- o Syntax for an SQL query in a POST request

```
POST _opendistro/_sql
{
  "query": "SELECT * FROM <your_index> LIMIT 50"
}
```

- o Syntax for an SQL query with the CSV format specified for the data to return

```
POST _opendistro/_sql?format=csv
{
  "query": "SELECT * FROM <your_index> LIMIT 50"
}
```

 Note

- For more information about SQL syntax, see [SQL](#).
- If you want to execute Java code by using SQL statements, download the JDBC driver at <https://github.com/opendistro-for-elasticsearch/sql-jdbc>.

11.4.3. Use the bsearch_querybuilder plug-in to construct query conditions

This topic describes how to use the bsearch_querybuilder plug-in for Elasticsearch to construct query conditions in Kibana. This plug-in implements visualized data queries. Only Kibana V6.7.0 and V6.8.2 support this plug-in.

Prerequisites

The bsearch_querybuilder plug-in is installed. You can contact O&M personnel to install the plug-in. For more information, see [Elasticsearch Operations and Maintenance Guide](#).

Procedure

1. Log on to the Kibana console of your Elasticsearch cluster. For more information, see [Log on to the Kibana console](#).
2. In the left-side navigation pane, click **Discover**. On the page that appears, click **Query** in the upper-right corner.

 Notice

- Before you create a query, make sure that you have created an index pattern. To create an index pattern in the Kibana console, click **Management** in the left-side navigation pane. On the page that appears, click **Index Patterns** in the Kibana section. Then, click **Create index pattern** and create an index pattern as prompted.
- If the bsearch_querybuilder plug-in is installed, but **Query** is not displayed in the Kibana console, the plug-in may not take effect. In this case, you must run the `kubectl delete deployments elasticsearch-cluster-kibana -n <your_es_namespace>` command on the relevant Kubernetes server to restart Kibana.

3. On the page that appears, specify query conditions and filters. Click the  icon to add a query condition. Click the  icon to add a filter for the query condition. Click the  icon to delete a query condition or filter.

The following figure shows a query example. In this example, the user condition is set to kimchy, and the type filters are set to tweet and Elasticsearch.

The screenshot shows the Kibana Query Builder interface. At the top, there are tabs for 'New', 'Save', 'Open', 'Share', 'Inspect', 'Query', and 'Auto-refresh'. The main query builder area shows a query structure: 'user' with a '(Match)' filter set to 'kimchy'. Below this, there is an 'OR' clause containing two 'message' filters, both with '(Match)' filters. The first 'message' filter is set to 'tweet', and the second is set to 'Elasticsearch'. A 'submit' button is located below the query builder. Below the query builder, there is a search bar with the text '> Search... (e.g. status:200 AND extension:PHP)' and a 'Refresh' button. At the bottom, there is a 'querybuilder' section with a 'Add a filter +' button and an 'Actions' dropdown. The results section shows two hits from the 'twitter' index. The first hit has a message: 'Another tweet, will it be indexed?' and a score of 1.575. The second hit has a message: 'Trying out Elasticsearch, so far so good?' and a score of 1.575.

4. Click submit.

11.4.4. Use alerting plug-ins to implement alerting for an Elasticsearch cluster

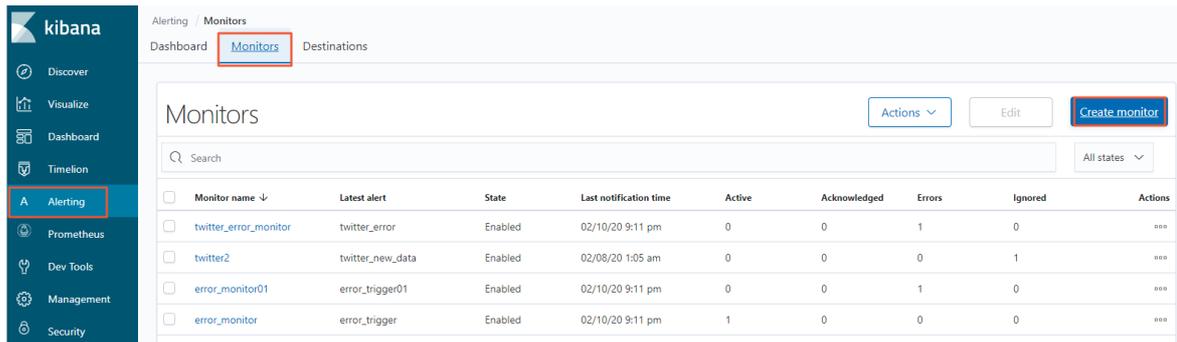
This topic describes how to use the alerting plug-ins of Elasticsearch and Kibana to implement alerting for an Elasticsearch cluster. You can configure alerts and customize alert thresholds and conditions for the cluster. The system sends alert notifications if the alert conditions are met.

Prerequisites

The `opendistro-alerting` plug-in for Elasticsearch and the `opendistro-alerting` plug-in for Kibana are installed. You can contact O&M personnel to install the plug-ins. For more information, see [Elasticsearch Operations and Maintenance Guide](#).

Procedure

1. Log on to the Kibana console of your Elasticsearch cluster. For more information, see [Log on to the Kibana console](#).
2. In the left-side navigation pane, click **Alerting**.
3. On the page that appears, click the **Monitors** tab.
4. In the upper-right corner of the tab, click **Create Monitor**.



5. On the page that appears, specify the required parameters.

Configure Monitor

Monitor name

Schedule

When do you want this monitor to run?

Frequency

By interval

Every

1 Minutes

Monitor state

Disabled monitors do not run.

Disable monitor

In this example, the name of the monitor is `error_monitor01`, and the monitor runs once a minute.

6. Define the monitor. Elasticsearch allows you to use visual images or specific query statements to define a monitor.
- If you want to use visual images to define the monitor, you must specify an **index** and a **time field**.
 - If you want to use specific query statements to define the monitor, specify an **index** and query conditions. Then, click **Run** and view results. This method is used in this topic.

The following figure shows the query condition. This query is to obtain data that includes the error keyword from messages.

Define Monitor Run

How do you want to define the monitor?

Index

You can use a * as a wildcard in your index pattern

Define extraction query

```

1 {
2   "size": 1,
3   "query": {
4     "match": {
5       "message": "error"
6     }
7   }
8 }
    
```

Extraction query response

```

1 {
2   "_shards": {
3     "total": 5,
4     "failed": 0,
5     "successful": 5,
6     "skipped": 0
7   },
8   "hits": [
9     {
10      "_index": "twitter",
11      "_type": "_doc",
12      "_source": {
13        "post_date": "2020-02-07T16:06:00",
14        "message": "alert: site error 404",
15        "user": "ouyangchucui"
16      },
17      "_id": "5",
18      "_score": 1.1001158
19    }
20  ],
21  "total": 2,
22  "max_score": 1.1001158
23 },
24 "took": 5,
25 "timed_out": false
26 }
    
```

```

{
  "size": 1,
  "query": {
    "match": {
      "message": "error"
    }
  }
}
    
```

7. Click **Create**.
8. On the **Create Trigger** page, define and configure an alert trigger.

Create Trigger

Define Trigger

Trigger name

Trigger names must be unique. Names can only contain letters, numbers, and special characters.

Severity level

Severity levels help you organize your triggers and actions. A trigger with a high severity level might page a specific individual, whereas a trigger with a low severity level might email a list.

Extraction query response

```

1 {
2   "_shards": {
3     "total": 5,
4     "failed": 0,
5     "successful": 5,
6     "skipped": 0
7   },
8   "hits": {
9     "hits": [],
10    "total": 2,
11    "max_score": 0
12  },
13  "took": 3,
14  "timed_out": false
15 }
    
```

Trigger condition [Info](#)

```

1 ctx.results[0].hits.total > 0
    
```

Parameter	Description
Trigger name	The name of the trigger.
Severity level	The severity of the trigger.
Extraction query response	The query response. This is specified in the alert definition and cannot be modified.
Trigger condition	The condition to trigger the alert. You can modify the condition based on your business requirements. For example, the condition is <code>ctx.results[0].hits.total > 0</code> . This indicates that an alert is reported when the number of returned entries is greater than 0.

9. Configure an alert notification. In the **Configure Actions** section, specify the required parameters.

The screenshot shows the 'Configure Actions' interface. At the top right is an 'Add action' button. Below it, a dropdown menu shows 'Custom webhook: notification_error' with a 'Delete' button. The 'Action name' field contains 'notification_error'. Below it, a note states 'Names can only contain letters, numbers, and special characters'. The 'Destination name' dropdown is set to 'dingding - (Custom webhook)'. A 'Message' field contains a template: 'Monitor {{ctx.monitor.name}} just entered alert status. Please investigate the issue. - Trigger: {{ctx.trigger.name}} - Severity: {{ctx.trigger.severity}} - Period start: {{ctx.periodStart}} - Period end: {{ctx.periodEnd}}'. A 'Send test message' link is at the bottom right. A 'Message preview' section shows the rendered message: 'Monitor error_monitor01 just entered alert status. Please investigate the issue. - Trigger: error_trigger01'.

Parameter	Description
Action name	The name of the notification.
Destination name	The destination address of the notification. In most cases, the address is a URL.
Message	The content of the notification. The content can be modified based on your requirements.
Message preview	The notification preview. This cannot be modified.

10. Click **Create**.

11. On the Alerting page, click the **Dashboard** tab to view the newly created alert.

Alerting / Dashboard

Dashboard Monitors Destinations

Alerts Acknowledge

Search All severity levels All alerts

<input type="checkbox"/> Alert start time ↓	Alert end time	Monitor name	Trigger name	Severity	State	Time acknowledged
<input checked="" type="checkbox"/> 02/08/20 12:00 am	02/08/20 1:06 am	twitter2	twitter_new_data	3	Completed	-
<input type="checkbox"/> 02/07/20 7:27 pm	-	error_monitor	error_trigger	1	Active	-
<input checked="" type="checkbox"/> 02/07/20 4:22 pm	-	twitter_monitor	twitter_error	1	Error	-