ALIBABA CLOUD

Alibaba Cloud

Apsara Stack Enterprise

ApsaraDB for RDS

ApsaraDB RDS for MySQL User Guide

Product Version: v3.16.2

Document Version: 20220913

Alibaba Cloud

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 `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.What is ApsaraDB RDS?

ApsaraDB RDS is a stable, reliable, and scalable online database service. Based on the high-performance storage, ApsaraDB RDS provides a set of solutions for disaster recovery, backup, restoration, monitoring, and migration.

ApsaraDB RDS supports four database engines: MySQL, SQL Server, PolarDB, and PostgreSQL. You can create database instances based on these database engines to meet your business requirements.

ApsaraDB RDS for MySQL

ApsaraDB RDS for MySQL is developed based on AliSQL and provides excellent performance. ApsaraDB RDS for MySQL is a tried and tested solution that handled the high-volume concurrent traffic during Double 11. ApsaraDB RDS for MySQL supports deployment with mixed x86 and ARM clusters. It integrates basic features such as allowlist configuration, backup and restoration, Transparent Data Encryption (TDE), data migration, and management for instances, accounts, and databases. ApsaraDB RDS for MySQL also provides the following advanced features:

- **Read-only instance:** In scenarios where ApsaraDB RDS for MySQL handles a small number of write requests but a large number of read requests, you can create read-only instances to scale up reading capabilities and increase the application throughput.
- **Read/write splitting:** The read/write splitting feature provides a read/write splitting endpoint. This endpoint enables automatic read/write splitting for a primary instance and all of its read-only instances. An application can connect to the read/write splitting endpoint to read and write data. Write requests are routed to the primary instance while read requests are routed to read-only instances based on their weights. To scale up reading capabilities of the system, you need only to add more read-only instances.

2.Log on to the ApsaraDB RDS console

This topic describes how to log on to the ApsaraDB RDS console.

Prerequisites

- Before you log on to the Apsara Uni-manager Management Console, the endpoint of the console is obtained from the deployment staff.
- We recommend that you use Google Chrome.

Procedure

1. Enter your username and password.

   Obtain the username and password that you use to log on to the Apsara Uni-manager Management Console from the operations administrator.

   > **Note** The first time that you log on to the Apsara Uni-manager Management Console, you must change the password of your account. For security purposes, your password must meet the minimum complexity requirements. The password must be 10 to 32 characters in length and must contain at least two of the following character types:
   >
   > - Uppercase or lowercase letters
   > - Digits
   > - Special characters, including exclamation points (!), at signs (@), number signs (#), dollar signs ($), and percent signs (%)

2. Click **Log On**.
3. If your account has multi-factor authentication (MFA) enabled, perform corresponding operations in the following scenarios:
   - You log on to the Apsara Uni-manager Management Console for the first time after MFA is enabled by the administrator:
     a. On the Bind Virtual MFA Device page, bind an MFA device.
     b. Enter the username and password again as in Step 2 and click **Log On**.
     c. Enter a six-digit MFA verification code and click **Authenticate**.
   - You have enabled MFA and bound an MFA device:

     Enter a six-digit MFA verification code and click **Authenticate**.

     > **Note** For more information, see the *Bind a virtual MFA device to enable MFA* topic in *Apsara Uni-manager Management Console User Guide*.
4. In the top navigation bar, choose **Products > Database Services > ApsaraDB RDS**.

3.Quick start

3.1. Limits

To ensure instance stability and security, ApsaraDB RDS for MySQL has specific service limits, as listed in the following table.

Item	Limits
Instance parameters	Instance parameters can be modified by using the ApsaraDB RDS console or API operations. Due to security and stability considerations, only specific parameters can be modified.
Root permissions of databases	The root or system administrator permissions are not provided.
Database backup	• Logical backup can be performed by using the CLI or GUI. • Physical backup can be performed only by using the ApsaraDB RDS console or API operations.
Database restoration	• Logical restoration can be performed by using the CLI or GUI. • Physical restoration can be performed only by using the ApsaraDB RDS console or API operations.
ApsaraDB RDS for MySQL storage engine	Only InnoDB is supported. • To ensure performance and security, we recommend that you use the InnoDB storage engine. • The TokuDB storage engine is not supported. Percona no longer provides support for TokuDB, which in extreme cases may lead to bugs that cannot be fixed and impact your workloads. • The MyISAM storage engine is not supported Due to the inherent shortcomings of the MyISAM engine, data may be lost. Only specific existing instances use the MylSAM engine. MyISAM engine tables in newly created instances are automatically converted to InnoDB engine tables. • The Memory engine is not supported. Newly created Memory tables are automatically converted into InnoDB tables.
Database replication	ApsaraDB RDS for MySQL provides a primary/secondary replication architecture. The secondary instances in this replication architecture are hidden and cannot be directly accessed.
Instance restart	Instances must be restarted by using the ApsaraDB RDS console or API operations.
Account and database management	ApsaraDB RDS for MySQL allows you to manage accounts and databases by using the ApsaraDB RDS console. ApsaraDB RDS for MySQL also allows you to create a privileged account to manage users, passwords, and databases.

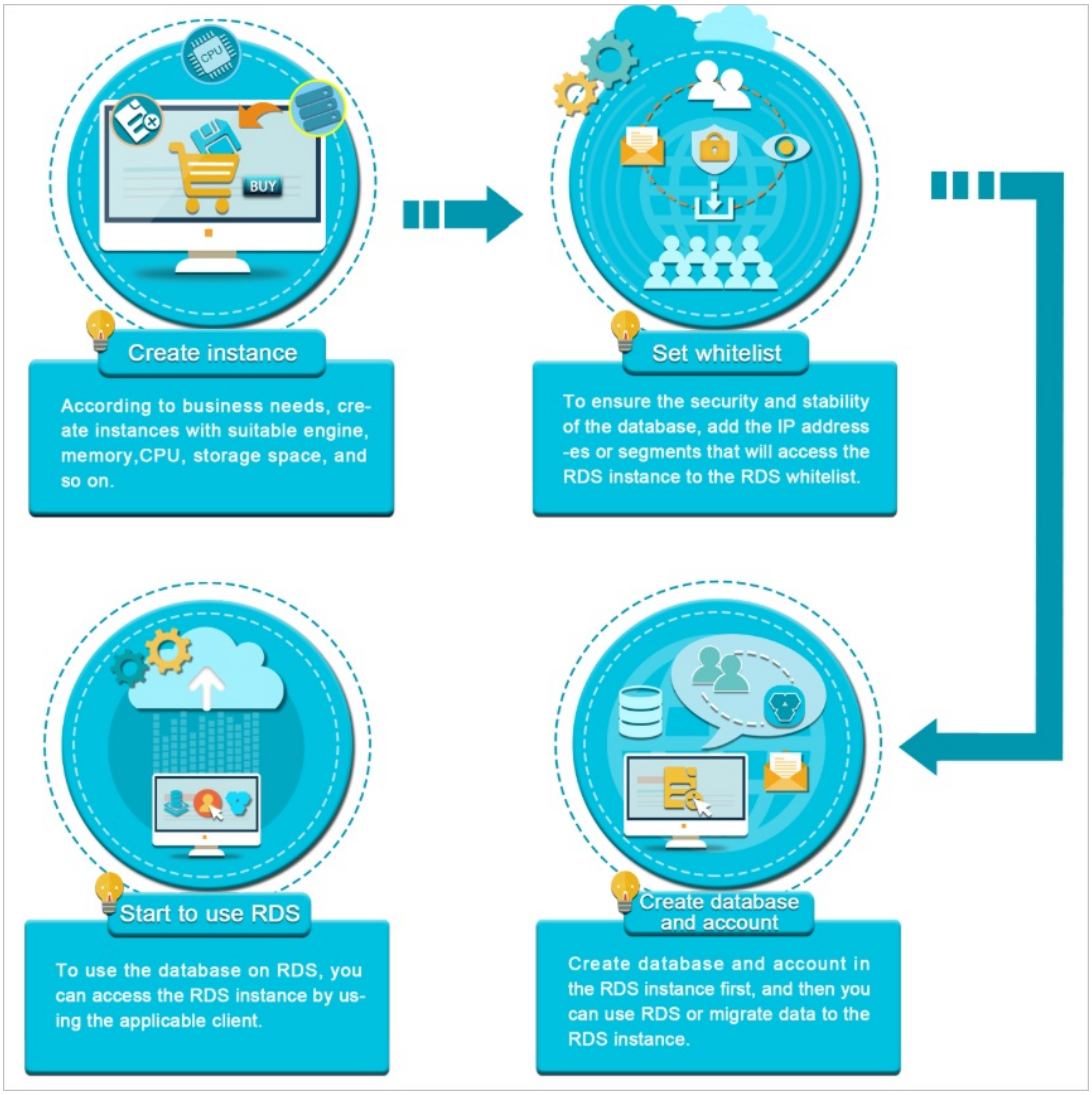
Item	Limits
Standard account	• Authorization is not allowed. • The ApsaraDB RDS console allows you to manage accounts and databases. • Instances that support standard accounts also support privileged accounts.
Privileged account	• Authorization is allowed on standard accounts. • The privileged account cannot be reverted to a standard account.

3.2. Procedure

ApsaraDB RDS quick start covers the following operations: creating an instance, configuring a whitelist, creating a database, creating an account, and connecting to the instance. This topic describes how to use ApsaraDB RDS and provides all the necessary information to create an ApsaraDB RDS instance. ApsaraDB RDS for MySQL is used in the example.

Typically, after an instance is created, you must perform several operations to make the instance ready for use, as shown in Quick start flowchart.

Quick start flowchart



- Create an instance

  An instance is a virtual database server on which you can create and manage multiple databases.

- Configure a whitelist

  After you create an ApsaraDB RDS instance, you must configure its whitelist to allow access from external devices.

  Whitelists make your ApsaraDB RDS instance more secure. We recommend that you maintain whitelists on a regular basis. The whitelist configuration process does not affect the normal operations of the ApsaraDB RDS instance.

- Create a database and Create an account

  Before you use a database, you must first create the database and an account in the ApsaraDB RDS instance.

- Connect to an ApsaraDB RDS for MySQL instance

  After you create an ApsaraDB RDS instance, configure a whitelist, and create a database and an account, you can connect to the instance by using a database client.

3.3. Create an instance

This topic describes how to create one or more instances in the ApsaraDB RDS console.

Prerequisites

An Apsara Stack tenant account is created.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, click **Create Instance** in the upper-right corner.
3. Select an option from the **Service Catalog** drop-down list and click **OK**.
4. Configure the parameters described in the following table.

Section	Parameter	Description
Basic Configurations	**Organization**	The organization to which the instance belongs.
	Resource Set	The resource set to which the instance belongs.
Area	**Region**	The region in which you want to create the instance. Services in different regions cannot communicate over an internal network. After the instance is created, the region cannot be changed.
	Primary Node Zone	The zone in which the primary instance is deployed. **Note** To deploy a zone-disaster recovery instance or an Enterprise Edition instance, select a zone whose name contains `MAZ`.
	Deployment Method	Specifies whether to deploy the primary and secondary instances in separate zones. ApsaraDB RDS supports **Multi-zone Deployment** and **Single-zone Deployment**. A deployment method is automatically selected based on the value of **Primary Node Zone**.
	Database Engine	The database engine of the instance. Select **MySQL**.
	Chip Architecture	The chip architecture of the host on which the instance is deployed. **Note** If you do not have permissions to select an option, contact the operations administrator to grant such permissions to your account.

Section	Parameter	Description
Specifications	Engine Version	The version of the database engine. Valid values: ○ 8.0 ○ 5.7 Note To deploy an Enterprise Edition instance, select 5.7.
	Edition	The edition of the instance. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*. Note To deploy an Enterprise Edition instance, select Enterprise Edition.
	Storage Type	The storage type of the instance. Select Local SSD.
	Instance Specifications	The instance specifications of the instance. The maximum number of connections and the maximum IOPS vary based on the memory size. The actual specifications are displayed in the console. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*.
	Storage Capacity	The storage capacity that is provided to store data files, system files, binlog files, and transaction files in the instance. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*.
	Connection Type	The connection type of the instance. Valid values: ○ **Internet**: Instances of this connection type can be connected over the Internet. ○ **Internal Network**: Instances of this connection type can be connected over an internal network. Note After the instance is created, the value of this parameter cannot be changed. Proceed with caution.
	Network Type	The network type of the instance. Valid values: ○ **Classic Network**: Cloud services in the classic network are not isolated. Unauthorized access to a cloud service is blocked only by the security group or allowlist policy of the service. Note Instances that use standard SSDs cannot be deployed in the classic network. ○ **VPC**: A virtual private cloud (VPC) helps you build an isolated network environment on Apsara Stack. You can customize route tables, CIDR blocks, and gateways in a VPC. We recommend that you select VPC for improved security.

Network Section	Parameter	Description
	VPC	The VPC in which you want to create the instance. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.
	vSwitch Zone	The zone in which the vSwitch is located. If you set **Primary Node Zone** to a zone whose name contains MAZ, you must specify the zone in which the vSwitch that you want to use is located. **Note** This parameter is available only when **Network Type** is set to **VPC**.
	VSwitch	The vSwitch to which the instance is connected. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.
	IP Whitelist	The IP address or CIDR block that is allowed to connect to the instance.
Instance Settings	Quantity	The number of instances that you want to create. Default value: 1.
	Instance Name	The name of the instance. ◦ The name must be 2 to 64 characters in length. ◦ The name must start with a letter. ◦ The name can contain letters, digits, hyphens (-), underscores (_), and colons (:). ◦ The name cannot start with http:// or https://.

5. Click **Submit**.

3.4. Initialization settings

3.4.1. Configure an IP address whitelist for an ApsaraDB RDS instance

After you create an ApsaraDB RDS instance, you must add the IP addresses or CIDR blocks that are used for database access to the IP address whitelist of the instance to ensure database security and reliability.

Context

IP address whitelists make your ApsaraDB RDS instance more secure and do not interrupt the operations of your ApsaraDB RDS instance when you configure whitelists. We recommend that you maintain your IP address whitelists on a regular basis.

To configure a whitelist, you can perform the following operations:

- Configure an IP address whitelist: Add IP addresses to allow them to connect to the ApsaraDB RDS instance.
- Configure an Elastic Compute Service (ECS) security group: Add an ECS security group for the ApsaraDB RDS instance to allow ECS instances in the group to connect to the ApsaraDB RDS instance.

Precautions

- The default IP address whitelist can be modified or cleared, but cannot be deleted.
- You can add up to 1,000 IP addresses or CIDR blocks to a whitelist. If you want to add a large number of IP addresses, we recommend that you merge them into CIDR blocks, such as 192.168.1.0/24.

Introduction to IPv6

IPv4 addresses are widely used, but the limited number of IPv4 addresses restricts the development of the Internet. Compared with IPv4 addresses, IPv6 addresses are more sufficient and allow more types of devices to access the Internet. ApsaraDB RDS supports both IPv4 and IPv6 addresses.

The following table describes the differences between IPv4 and IPv6.

Item	IPv4	IPv6
Address length	32 bits (4 bytes)	128 bits (16 bytes)
Number of addresses	2^32	2^128
Address format	xxx.xxx.xxx.xxx Where xxx is a decimal number that can range from 0 to 255. Each x is a decimal integer, and leading zeros can be omitted. Example: 192.168.1.1	xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx Where each x is a hexadecimal number, and leading zeros can be omitted. You can use a double colon (::) once in an IPv6 address to indicate a series of zeros. Example: CDDC:0000:0000:0000:8475:1111:3900:2020
Address Resolution Protocol (ARP)	Uses broadcast ARP Request frames to resolve an IP address to a link layer address.	Uses multicast neighbor solicitation messages to resolve an IP address to a link layer address.
Security	Implements a security mechanism based on applications and cannot provide protections at the IP layer.	Supports packet fragmentation to ensure data confidentiality and integrity and provides security at the IP layer.
LAN connection	Connects to LANs by using network interfaces.	Can work with Ethernet adapters and is supported over virtual Ethernet networks between logical partitions.

Item	IPv4	IPv6
Address type	• Unicast address • Multicast address • Broadcast address	• Unicast address • Multicast address • Anycast address

Create an IP address whitelist

Each IP address whitelist of an ApsaraDB RDS instance can contain **IPv4** or **IPv6** addresses. By default, the system provides an IP address whitelist of the **IPv4** type. If you want an IP address whitelist of the **IPv6** type, manually create one.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. On the **Whitelist Settings** tab, click **Create Whitelist**. In the dialog box that appears, configure the following parameters.

Parameter	Description
Whitelist Name	The name of the IP address whitelist. **Note** ◦ The name can contain lowercase letters, digits, and underscores (_). ◦ The name must start with a lowercase letter and end with a lowercase letter or digit. ◦ The name must be 2 to 32 characters in length.
IP Type	The IP type of the IP address whitelist. Valid values: ◦ **IPv4** ◦ **IPv6** **Note** For more information about the differences between IPv4 and IPv6, see the "Introduction to IPv6" section of this topic.
IP Addresses	The IP addresses that are allowed to access the instance.

Configure an IP address whitelist

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. On the **Whitelist Settings** tab, click **Edit** corresponding to an IP address whitelist.

> **Note** If you want to connect an ECS instance to an ApsaraDB 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.

5. In the **Edit Whitelist** dialog box, enter the IP addresses or CIDR blocks that are allowed to access the instance and click **OK**.

   > **Note**
   > - Limits for **IPv4** addresses:
   >   - You must separate multiple IP addresses with commas (,). A maximum of 1,000 different IP addresses can be added.
   >
   >   Supported formats are `0.0.0.0/0`, IP addresses such as `10.23.12.24`, or CIDR blocks such as `10.23.12.24/24`. `/24` indicates the length of the IP address prefix in the CIDR block. An IP address prefix can contain 1 to 32 bits.
   >   - If an IP address whitelist is empty or contains `0.0.0.0/0`, all IP addresses can access the ApsaraDB RDS instance. This may cause security risks to the instance. Proceed with caution.
   > - Limits for **IPv6** addresses:
   >   - You must separate multiple IP addresses with commas (,). A maximum of 1,000 different IP addresses can be added.
   >
   >   Supported formats are `::`, IP addresses such as `0:0:0:0:0:0:0:1`, or CIDR blocks such as `0:0:0:0:0:0:0:1/24`. `/24` indicates the length of the IP address prefix in the CIDR block. An IP address prefix can contain 1 to 128 bits.
   >   - If an IP address whitelist is empty or contains only `::`, all IP addresses can access the ApsaraDB RDS instance. This may cause security risks to the instance. Proceed with caution.
   > - You cannot specify both **IPv4** and **IPv6** addresses in a single IP address whitelist. If you want to specify both IPv4 and IPv6 addresses, specify them in separate IP address whitelists.
   > - If you click **Add Internal IP Addresses of ECS Instances**, the IP addresses of all the ECS instances that are created within your Apsara Stack tenant account appear. Then, you can select the IP addresses and add them to an IP address whitelist.

3.4.2. Create an account

After you create an ApsaraDB RDS instance and configure its IP address whitelist, you must create a database and an account on the instance. This topic describes how to create privileged and standard accounts.

Context

ApsaraDB RDS for MySQL supports two types of database accounts: privileged and standard. You can manage all your accounts and databases in the ApsaraDB RDS console. For more information about permissions that can be granted to each type of account, see Account permissions.

Account type	Description
Privileged account	• You can create and manage privileged accounts by using the ApsaraDB RDS console or API operations. • You can create only a single privileged account on each ApsaraDB RDS instance. The privileged account can be used to manage all standard accounts and databases on the instance. • A privileged account allows you to manage permissions to a fine-grained level. For example, you can grant each standard account the permissions to query specific tables. • A privileged account has the permissions to disconnect all standard accounts on the instance.
Standard account	• You can create and manage standard accounts by using the ApsaraDB RDS console, API operations, or SQL statements. • You can create up to 500 standard accounts on an instance. • You must manually grant standard accounts the specific database permissions. • You cannot use a standard account to create, manage, or disconnect other accounts from databases.

Account type	Maximum number of databases	Maximum number of tables	Maximum number of accounts
Privileged account	Unlimited	< 200,000	Varies based on the engine parameter settings of the instance.
Standard account	500	< 200,000	Varies based on the engine parameter settings of the instance.

Create a privileged account

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.
4. On the **Accounts** tab, click **Create Account**.
5. On the **Create Account** page, configure the following parameters.

Parameter	Description
Database Account	Enter the name of the account. The account name must meet the following requirements: ◦ The name is 1 to 16 characters in length. ◦ The name starts with a lowercase letter and ends with a lowercase letter or digit. ◦ The name contains lowercase letters, digits, and underscores (_).

Parameter	Description
Account Type	Select Privileged Account.
Password	Enter the password of the account. The password must meet the following requirements: ○ The password is 8 to 32 characters in length. ○ The password contains at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ○ Special characters include ! @ # $ % ^ & * ( ) _ + - =
Re-enter Password	Enter the password of the account again.
Description	Optional. Enter information about the account to facilitate subsequent management. The description can be up to 256 characters in length.

6. Click **Create**.

Reset the permissions of a privileged account

If an issue occurs on the privileged account, you can enter the password of the privileged account to reset permissions. For example, you can reset the permissions if the permissions are unexpectedly revoked.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.
4. On the **Accounts** tab, find the privileged account and click **Reset Permissions** in the **Actions** column.
5. On the **Initialize Account** page, enter the password of the privileged account and click **OK**.

Create a standard account

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.
4. On the **Accounts** tab, click **Create Account**.
5. On the **Create Account** page, configure the following parameters.

Parameter	Description

Parameter	Description
Database Account	Enter the name of the account. The account name must meet the following requirements: ○ The name is 1 to 16 characters in length. ○ The name starts with a lowercase letter and ends with a lowercase letter or digit. ○ The name contains lowercase letters, digits, and underscores (_).
Account Type	Select Standard Account.
Authorized Databases	Select one or more databases on which you want to grant permissions to the account. You can also leave this parameter empty at this time and authorize databases after the account is created. i. Select one or more databases from the Unauthorized Databases section and click **Add** to add them to the Authorized Databases section. ii. In the Authorized Databases section, select the **Read/Write**, **Read-only**, **DDL Only**, or **DML Only** permissions on each authorized database. If you want to grant the same permissions on multiple databases to the account, click the button in the upper-right corner of the section. The button may appear as **Set All to Read/Write**.
Password	Enter the password of the account. The password must meet the following requirements: ○ The password is 8 to 32 characters in length. ○ The password contains at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ○ Special characters include ! @ # $ % ^ & * ( ) _ + - =
Re-enter Password	Enter the password of the account again.
Description	Optional. Enter information about the account to facilitate subsequent management. The description can be up to 256 characters in length.

6. Click **Create**.

Account permissions

Account type	Authorization type	Permission				
Privileged account	None	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	RELOAD	PROCESS	REFERENCES	INDEX
		ALTER	CREATE TEMPORARY TABLES	LOCK TABLES	EXECUTE	REPLICATION SLAVE
		REPLICATION CLIENT	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		CREATE USER	EVENT	TRIGGER	None	None
Standard account	**Read-only**	SELECT	LOCK TABLES	SHOW VIEW	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT	None	None	None	None
	Read/write	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	REFERENCES	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	EXECUTE	CREATE VIEW	SHOW VIEW	CREATE ROUTINE
		ALTER ROUTINE	EVENT	TRIGGER	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT	None	None	None	None
	DDL-only	CREATE	DROP	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT	None	None
		SELECT	INSERT	UPDATE	DELETE	CREATE TEMPORARY TABLES

Account type	Authorization type	Permission				
		LOCK TABLES	EXECUTE	SHOW VIEW	EVENT	TRIGGER
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT	None	None

3.4.3. Create a database

After you create an ApsaraDB RDS instance and configure its IP address whitelist, you must create a database and an account on the instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Databases**.
4. Click **Create Database**. On the page that appears, configure the following parameters.

Parameter	Description
Database Name	○ The name must be 1 to 64 characters in length. ○ The name must start with a letter and end with a letter or a digit. ○ The name can contain lowercase letters, digits, underscores (_), and hyphens (-). ○ The name must be unique within the instance.
Supported Character Sets	Select **utf8**, **gbk**, **latin1**, **utf8mb4**, or **all**. If you want to use other character sets, select **all**, and then select the required character set from the list.
Description	Optional. Enter information about the database to facilitate subsequent management. The description must be 2 to 256 characters in length.

5. Click **Create**.

3.5. Connect to an ApsaraDB RDS for MySQL instance

After you complete the initial configuration of your ApsaraDB RDS for MySQL instance, you can connect to it from an Elastic Compute Service (ECS) instance or an on-premises client.

Context

After you perform operations such as Create an instance, Configure a whitelist, and Create an account, you can use a general database client or configure the endpoint, port number, and account information in an application to connect to the ApsaraDB RDS for MySQL instance.

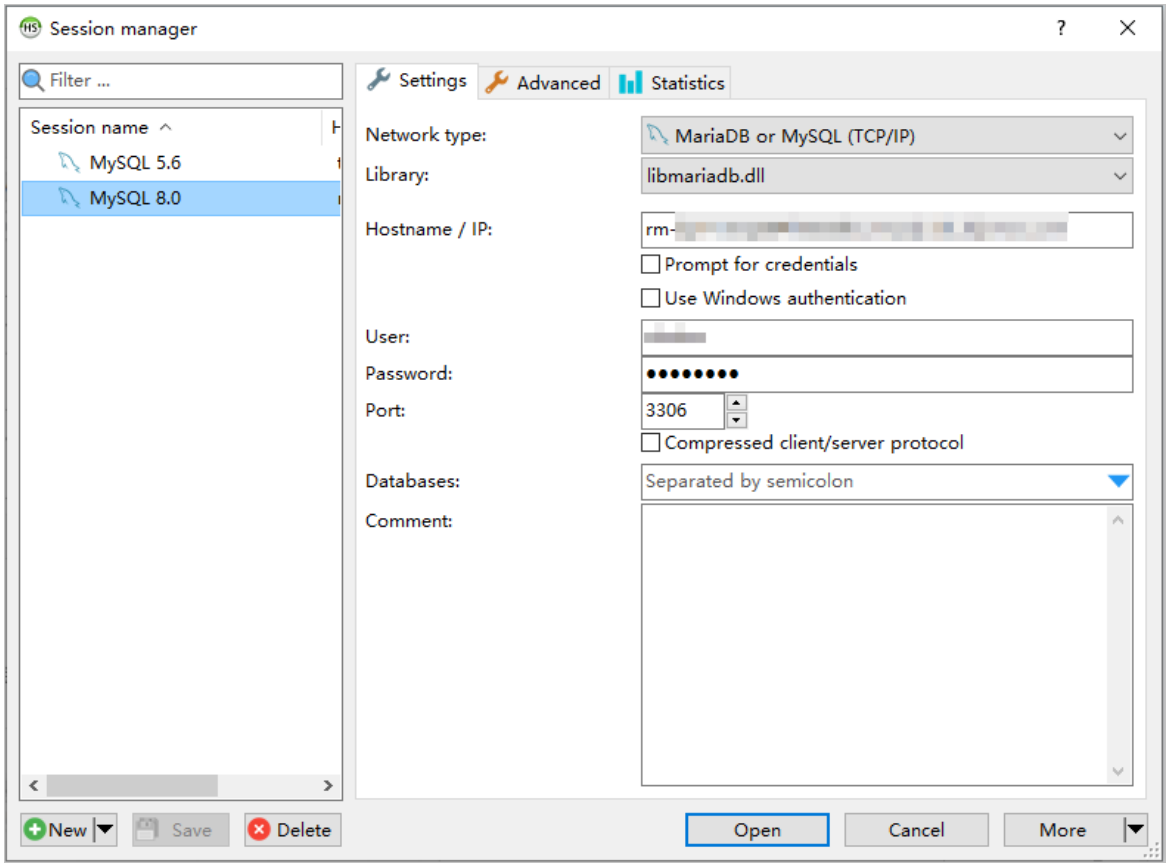
If you need to connect an ECS instance to an ApsaraDB RDS instance, you must make sure that both instances are in the classic network or in the same virtual private cloud (VPC), and the IP address of the ECS instance is added to an IP address whitelist of the ApsaraDB RDS instance.

Connect to an instance from a client

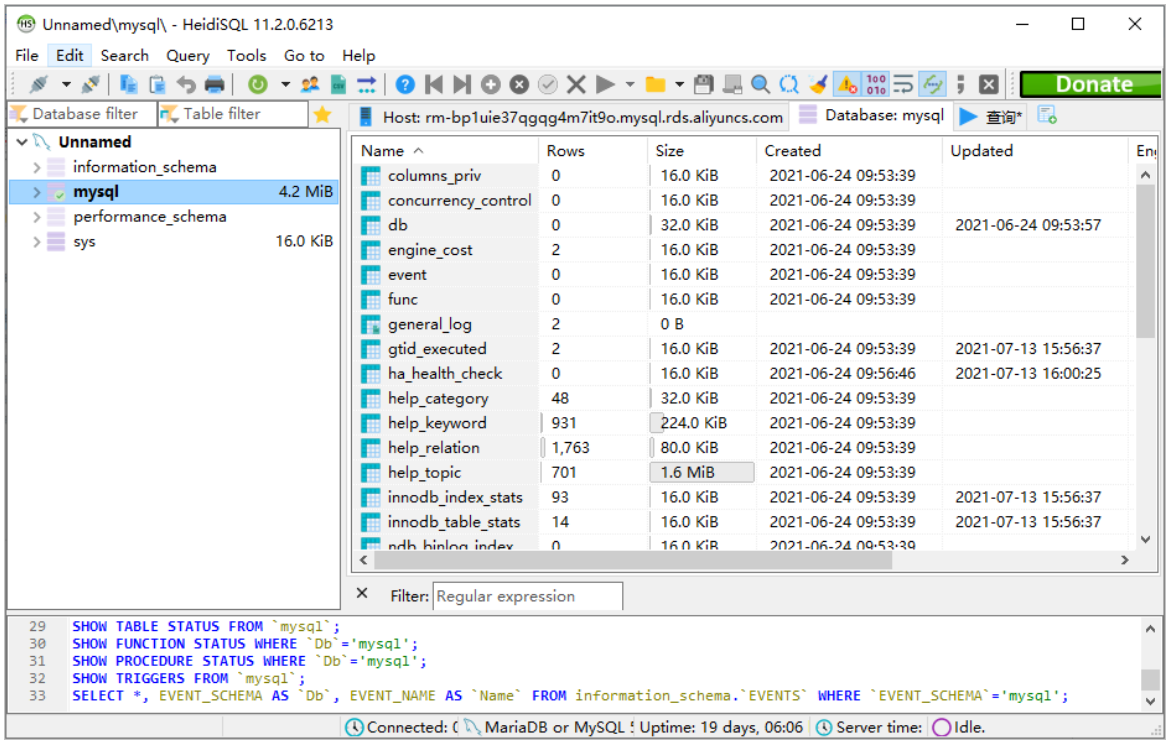
ApsaraDB RDS for MySQL is fully compatible with open source MySQL. You can connect to an ApsaraDB RDS instance from a database client by using a method similar to the method that is used to connect to an open source MySQL database. In the following example, the HeidiSQL client is used.

1. Start the HeidiSQL client.
2. In the lower-left corner of the Session manager dialog box, click **New**.
3. Enter information about the ApsaraDB RDS instance to which you want to connect. The following table describes the required parameters.

Parameter	Description
Network type	Select the network type of the ApsaraDB RDS instance to which you want to connect. For example, select **MariaDB or MySQL (TCP/IP)**.
Hostname / IP	Enter the internal or public endpoint of the ApsaraDB RDS instance. ○ If your client is deployed on an ECS instance that is in the same region and has the same network type as the ApsaraDB RDS instance, use the internal endpoint. For example, if the ECS and ApsaraDB RDS instances are both in a VPC located in the China (Hangzhou) region, you can use the internal endpoint of the ApsaraDB RDS instance to create a secure connection. ○ In other scenarios, use the public endpoint. To view the internal and public endpoints and port numbers of the ApsaraDB RDS instance, perform the following operations: i. Log on to the ApsaraDB for RDS console. ii. Find the ApsaraDB RDS instance to which you want to connect and click its ID. iii. In the **Basic Information** section, view the internal and public endpoints and port numbers of the instance.
User	Enter the name of the account used to connect to the ApsaraDB RDS instance.
Password	Enter the password of the account used to connect to the ApsaraDB RDS instance.
Port	If you connect to the instance over an internal network, enter the internal port number of the instance. If you connect to the instance over the Internet, enter the public port number of the instance.



4. Click **Open**. If the connection information is correct, you can connect to the instance.



4.Instances

4.1. Create an instance

This topic describes how to create one or more instances in the ApsaraDB RDS console.

Prerequisites

An Apsara Stack tenant account is created.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, click **Create Instance** in the upper-right corner.
3. Select an option from the **Service Catalog** drop-down list and click **OK**.
4. Configure the parameters described in the following table.

Section	Parameter	Description
Basic Configurations	**Organization**	The organization to which the instance belongs.
	Resource Set	The resource set to which the instance belongs.
Area	**Region**	The region in which you want to create the instance. Services in different regions cannot communicate over an internal network. After the instance is created, the region cannot be changed.
	Primary Node Zone	The zone in which the primary instance is deployed. **Note** To deploy a zone-disaster recovery instance or an Enterprise Edition instance, select a zone whose name contains `MAZ`.
	Deployment Method	Specifies whether to deploy the primary and secondary instances in separate zones. ApsaraDB RDS supports **Multi-zone Deployment** and **Single-zone Deployment**. A deployment method is automatically selected based on the value of **Primary Node Zone**.
	Database Engine	The database engine of the instance. Select **MySQL**.
	Chip Architecture	The chip architecture of the host on which the instance is deployed. **Note** If you do not have permissions to select an option, contact the operations administrator to grant such permissions to your account.

Section	Parameter	Description
Specifications	Engine Version	The version of the database engine. Valid values: ○ 8.0 ○ 5.7 Note To deploy an Enterprise Edition instance, select 5.7.
	Edition	The edition of the instance. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*. Note To deploy an Enterprise Edition instance, select Enterprise Edition.
	Storage Type	The storage type of the instance. Select Local SSD.
	Instance Specifications	The instance specifications of the instance. The maximum number of connections and the maximum IOPS vary based on the memory size. The actual specifications are displayed in the console. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*.
	Storage Capacity	The storage capacity that is provided to store data files, system files, binlog files, and transaction files in the instance. For more information, see **Instance types** in *ApsaraDB RDS Product Introduction*.
	Connection Type	The connection type of the instance. Valid values: ○ **Internet**: Instances of this connection type can be connected over the Internet. ○ **Internal Network**: Instances of this connection type can be connected over an internal network. Note After the instance is created, the value of this parameter cannot be changed. Proceed with caution.

Section	Parameter	Description
Network	Network Type	The network type of the instance. Valid values: ○ **Classic Network**: Cloud services in the classic network are not isolated. Unauthorized access to a cloud service is blocked only by the security group or allowlist policy of the service. **Note** Instances that use standard SSDs cannot be deployed in the classic network. ○ **VPC**: A virtual private cloud (VPC) helps you build an isolated network environment on Apsara Stack. You can customize route tables, CIDR blocks, and gateways in a VPC. We recommend that you select VPC for improved security.
	VPC	The VPC in which you want to create the instance. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.
	vSwitch Zone	The zone in which the vSwitch is located. If you set **Primary Node Zone** to a zone whose name contains MAZ, you must specify the zone in which the vSwitch that you want to use is located. **Note** This parameter is available only when **Network Type** is set to **VPC**.
	VSwitch	The vSwitch to which the instance is connected. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.
	IP Whitelist	The IP address or CIDR block that is allowed to connect to the instance.
Instance Settings	Quantity	The number of instances that you want to create. Default value: 1.
	Instance Name	The name of the instance. ○ The name must be 2 to 64 characters in length. ○ The name must start with a letter. ○ The name can contain letters, digits, hyphens (-), underscores (_), and colons (:). ○ The name cannot start with http:// or https://.

5. Click **Submit**.

4.2. View basic information of an instance

This topic describes how to view the details of an ApsaraDB RDS instance, such as its basic information, internal network connection information, status, and configurations.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. Use one of the following methods to go to the **Basic Information** page of an instance:
   - On the **Instances** page, click the ID of an instance to go to the **Basic Information** page.
   - On the **Instances** page, click **Manage** in the **Actions** column corresponding to an instance to go to the **Basic Information** page.

4.3. Restart an instance

This topic describes how to manually restart an ApsaraDB RDS instance. This applies if the number of connections exceeds the specified threshold or if an instance has performance issues.

Prerequisites

The instance is in the **Running** state.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. Click **Restart Instance** in the upper-right corner.

   > **Note** When you restart an instance, applications are disconnected from the instance. We recommend that you make appropriate service arrangements before you restart an instance. Proceed with caution.

4. In the Restart Instance message, click **Confirm**.

4.4. Change the specifications of an instance

This topic describes how to change the specifications of your instance, such as the instance type and storage capacity, if the specifications do not meet the requirements of your application.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the upper-right corner of the **Configuration Information** section, click **Change Specifications**.

4. On the **Change Specifications** page, set **Edition**, **Instance Type**, and **Storage Capacity**.
5. Click **Submit**.

4.5. Set a maintenance window

This topic describes how to set a maintenance window for an ApsaraDB RDS instance.

Context

To ensure the stability of ApsaraDB RDS instances, the backend system performs maintenance of the instances at irregular intervals. The default maintenance window is from 02:00 to 06:00. You can set the maintenance window to the off-peak period of your business to avoid impact on business.

Precautions

- To ensure stability of the maintenance process, the instance changes to the **Maintaining Instance** state before the maintenance window. When the instance is in this state, access to data in the database and query operations such as performance monitoring are not affected. However, except for account and database management and IP address whitelist configuration, modification operations such as upgrade, downgrade, and restart are temporarily unavailable.
- You may encounter a network interruption during the maintenance window. Make sure that your application is configured to automatically reconnect to the instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Configuration Information** section, click **Configure** to the right of **Maintenance Window**.
4. Select a maintenance window and click **Save**.

> **Note** The maintenance window is displayed in UTC+8.

4.6. Change the data replication mode

You can set the data replication mode between primary and secondary ApsaraDB RDS instances to improve database availability.

Context

ApsaraDB RDS supports the following data replication modes:

- Semi-sync

  After an application-initiated update is complete on the primary instance, logs are synchronized to all secondary instances. This transaction is considered committed after at least one secondary instance has received the logs, regardless of whether the secondary instance finishes executing the updates specified in the logs.

If the secondary instances are unavailable or a network exception occurs between the primary and secondary instances, semi-synchronous replication degrades to the asynchronous mode.

- Asynchronous

  When your application initiates a request to add, delete, or modify data, the primary instance responds to your application immediately after it completes the operation. At the same time, the primary instance starts to asynchronously replicate data to its secondary instances. During asynchronous data replication, the unavailability of secondary instances does not affect the operations on the primary instance. Data remains consistent even if the primary instance is unavailable.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Service Availability**.
4. In the upper-right corner of the **Availability Information** section, click **Change Data Replication Mode**.
5. In the dialog box that appears, select a data replication mode and click **OK**.

4.7. Release an instance

This topic describes how to manually release an instance.

Precautions

- Only instances in the running state can be manually released.
- After an instance is released, the instance data is immediately deleted. We recommend that you back up your data before you release an instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Actions** column corresponding to the instance you want to release, choose **More** > **Release Instance**.
4. In the message that appears, click **OK**.

4.8. Update the minor version of an instance

ApsaraDB RDS for MySQL supports automatic and manual updates of the minor version. These updates increase performance, provide new features, and fix known issues.

Introduction

By default, ApsaraDB RDS for MySQL automatically updates the minor version. You can log on to the ApsaraDB RDS console, go to the **Basic Information** page of your ApsaraDB RDS instance, and then view the current **Minor Version Upgrade Mode** in the Configuration Information section.

- **Auto**: When a new minor version is released, the system automatically updates the minor version of your instance during the specified maintenance window. For more information, see Set a maintenance window.
- **Manual**: You can manually update the minor version on the **Basic Information** page. For more information, see Manually update the minor version.

Precautions

- When you update the minor engine version of your ApsaraDB RDS instance, a network interruption of about 30 seconds may occur. We recommend that you update the minor engine version during off-peak hours or make sure that your application is configured to automatically reconnect to the instance.
- After you update the minor engine version of your ApsaraDB RDS instance, you cannot downgrade the instance version.
- After you upgrade the specifications of your ApsaraDB RDS instance, ApsaraDB RDS updates the instance to the latest minor engine version.

Change the minor version update mode

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Configuration Information** section of the Basic Information page, click **Configure** to the right of **Minor Version Upgrade Mode**.
4. In the dialog box that appears,select **Auto** or **Manual** and click **OK**.

> **Note** By default, the minor version update mode is set to **Auto**.

Manually update the minor version

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Configuration Information** section of the page, click **Upgrade Minor Engine Version**.

> **Note** The **Upgrade Minor Engine Version** button is displayed only when a new minor version is available.

4. In the dialog box that appears, specify the update time and click **OK**.

FAQ

- Q: After I updated the minor version of my ApsaraDB RDS instance, the MySQL version remains unchanged. Why?

  A: The minor version that you updated is the minor engine version of ApsaraDB RDS, but not the MySQL version. To view the minor version of your instance, you can execute the `show variables like '%rds_release_date%'` statement.

- Q: When an update takes effect, is my instance updated only to the next minor version?

  A: No, when an update takes effect, your instance is updated to the latest minor version.

4.9. Modify parameters of an instance

This topic describes how to view and modify the values of some parameters and query parameter modification records in the console.

Precautions

- To ensure instance stability, you can select specific parameters to modify in the ApsaraDB RDS console.
- When you modify parameters on the **Editable Parameters** tab, you can refer to the **Value Range** column corresponding to each parameter.
- After some parameters are modified, you must restart your ApsaraDB RDS instance for the changes to take effect. You can refer to the **Force Restart** column on the **Editable Parameters** tab. We recommend that you modify the parameters of an instance during off-peak hours and make sure that your applications are configured to automatically reconnect to your instance.

Modify parameters

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Parameters**.
4. Perform the following operations:

   Export the parameter settings of the ApsaraDB RDS instance to your computer.

   On the Editable Parameters tab, click **Export Parameters**. The parameter settings of the ApsaraDB RDS instance are exported as a TXT file to your computer.

   Modify and import the parameter settings.

   i. After you modify parameters in the exported parameter file, click **Import Parameters** and copy the parameter settings to the field.

   ii. Click **OK**.

   iii. In the upper-right corner of the page, click **Apply Changes**.

   > **Note**
   >
   > - If the new parameter value takes effect only after you restart your instance, the system prompts you to restart the ApsaraDB RDS instance. We recommend that you restart the ApsaraDB RDS instance during off-peak hours and make sure that your applications are configured to automatically reconnect to your instance.
   > - Before the new parameter values are applied, you can click **Cancel Changes** to cancel the modification.

   Modify a single parameter.

   i. On the **Editable Parameters** tab, find the parameter that you want to modify and click the icon in the **Actual Value** column.

   ii. Enter a new value based on the value range that is displayed.

   iii. Click **Confirm**.

iv. In the upper-right corner of the page, click **Apply Changes**.

> **Note**
>
> - If the new parameter value takes effect only after you restart your instance, the system prompts you to restart the ApsaraDB RDS instance. We recommend that you restart the ApsaraDB RDS instance during off-peak hours and make sure that your applications are configured to automatically reconnect to your instance.
> - Before the new parameter value is applied, you can click **Cancel Changes** to cancel the modification.

View the parameter modification history

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Parameters**.
4. On the Parameters page, click the **Edit History** tab.
5. Select a time range and click **Search**.

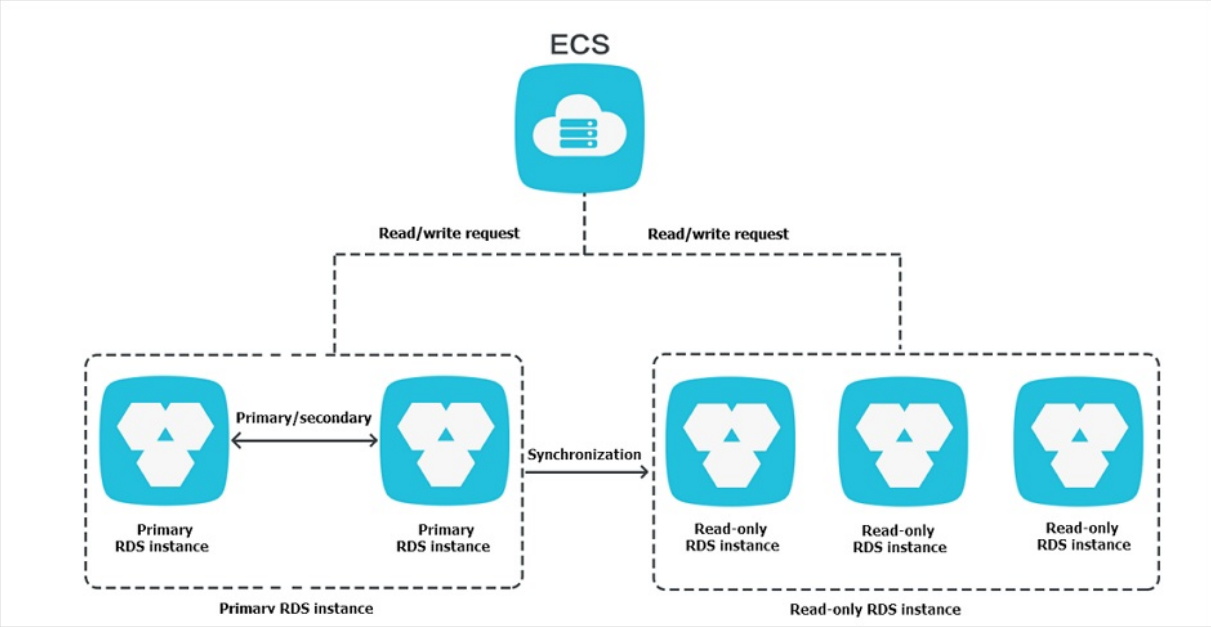
4.10. Read-only instances

4.10.1. Overview of read-only instances

ApsaraDB RDS for MySQL allows you to create read-only instances. In scenarios where an instance has a small number of write requests but a large number of read requests, you can create read-only instances to distribute database access loads away from the primary instance. This topic describes the features and limits of read-only instances.

To scale up the reading capability and distribute database access loads, you can create one or more read-only instances in a region. Read-only instances can increase the application throughput when a large amount of data is being read.

A read-only instance with a single physical node and no backup node uses the native replication capability of MySQL to synchronize changes from the primary instance to all its read-only instances. Read-only instances must be in the same region as the primary instance but do not have to be in the same zone as the primary instance. The following figure shows the topology of read-only instances.



Read-only instances have the following features:

- Specifications of a read-only instance can be different from those of the primary instance and can be changed at any time. This facilitates elastic scaling.
- Read-only instances do not require account or database maintenance. Account and database information is synchronized from the primary instance.
- The whitelists of read-only instances can be independently configured.
- System performance monitoring is provided.

  ApsaraDB RDS provides up to 20 system performance monitoring views, including those for disk capacity, IOPS, connections, CPU utilization, and network traffic. You can view the load of instances.
- ApsaraDB RDS provides a variety of optimization recommendations, such as storage engine check, primary key check, large table check, and check for excessive indexes and missing indexes. You can optimize your databases based on the optimization recommendations and specific applications.

4.10.2. Create a read-only instance

You can create read-only instances of different specifications based on your business requirements.

Precautions

- A maximum of five read-only instances can be created for a primary instance.
- Backup settings and temporary backup are not supported.
- Instance restoration is not supported.
- Data migration to read-only instances is not supported.
- Database creation and deletion are not supported.
- Account creation, deletion, authorization, and password changes are not supported.
- After a read-only instance is created, you cannot restore data by directly overwriting the primary instance with a backup set.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Distributed by Instance Role** section on the right side of the **Basic Information** page, click **Create Read-only Instance**.
4. On the **Create Read-only RDS Instance** page, configure the read-only instance parameters.

Section	Parameter	Description
Region	**Region**	The region in which you want to create the read-only instance.
Specifications	**Database Engine**	The database engine of the read-only instance, which is the same as that of the primary instance and cannot be changed.
	Engine Version	The version of the database engine, which is the same as that of the primary instance and cannot be changed.
	Edition	Set the value to **Read-only**.
	Instance Type	The instance type of the read-only instance. The instance type of the read-only instance can be different from that of the primary instance, and can be changed at any time to facilitate flexible upgrade and downgrade.
	Storage Capacity	The storage capacity of the read-only instance. To ensure sufficient I/O throughput for data synchronization, we recommend that you select at least the same instance type and storage capacity as the primary instance for the read-only instance. Valid values: 20 to 6000. Unit: GB. The value is in 1 GB increments.
Network Type	**Network Type**	The network type of the read-only instance. This must be the same as that of the primary instance and cannot be changed.
	VPC	The VPC in which you want to create the read-only instance.
	vSwitch	The vSwitch in the VPC.

5. Click **Submit**.

4.10.3. View details of read-only instances

This topic describes how to view details of read-only instances. You can go to the Basic Information page of a read-only instance from the Instances page or from the read-only instance list of the primary instance. Read-only instances are managed in the same manner as primary instances. The Basic Information page shows the management operations that can be performed.

View details of a read-only instance from the Instances page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, click the ID of a read-only instance. The **Basic Information** page appears.

> **Note** In the instance list, **Instance Role** of read-only instances is displayed as Read-only Instance.

View details of a read-only instance from the Basic Information page of the primary instance

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. On the **Basic Information** page, move the pointer over the number below **Read-only Instances** in the **Distributed by Instance Role** section. The ID of the read-only instance is displayed.
4. Click the ID of the read-only instance to go to the Basic Information page of the read-only instance.

4.11. Manage instances in the recycle bin

This topic describes how to recreate or destroy an ApsaraDB RDS instance in the recycle bin.

Description

All ApsaraDB RDS for MySQL instances, except for the following instances, are moved to the recycle bin after they are manually released:

- Instances that are manually released within seven days after they are created.
- **Read-only instances** and **Enterprise Edition instances**.

Recreate an instance

If you manually release an ApsaraDB RDS instance that runs for more than seven days, backup files of the instance are retained for another eight days. Within the eight-day retention period, you can create another instance and restore data from the backup files to the new instance. After the eight-day retention period elapses, the backup files of the instance are deleted.

1. Log on to the ApsaraDB for RDS console.
2. In the upper-left corner of the page, select an organization. In the left-side navigation pane, click **Locked Instances**.
3. Find the released instance and click **Rebuild** in the Actions column.
4. On the **Restore Instance** page, select **Storage Capacity** and **Network Type** for the new instance and click **Submit**.

> **Note**
>
> - If you set **Network Type** to **VPC**, you must configure a **virtual private cloud (VPC)** and a **vSwitch**.
> - Other configurations remain the same as the original instance and cannot be changed.

Destroy an instance

You can destroy an ApsaraDB RDS instance that is no longer needed in the recycle bin.

> **Warning** After you destroy an ApsaraDB RDS instance, **all** the backup files of the instance are destroyed, including regular data backup files, archived backup files, and log backup files. The destroyed backup files **cannot be restored**. Proceed with caution.

1. Log on to the ApsaraDB for RDS console.
2. In the upper-left corner of the page, select an organization. In the left-side navigation pane, click **Locked Instances**.
3. Find the released instance and click **Destroy** in the Actions column. In the message that appears, click **Confirm**.

5.Accounts

5.1. Create an account

After you create an ApsaraDB RDS instance and configure its IP address whitelist, you must create a database and an account on the instance. This topic describes how to create privileged and standard accounts.

Context

ApsaraDB RDS for MySQL supports two types of database accounts: privileged and standard. You can manage all your accounts and databases in the ApsaraDB RDS console. For more information about permissions that can be granted to each type of account, see Account permissions.

Account type	Description
Privileged account	• You can create and manage privileged accounts by using the ApsaraDB RDS console or API operations. • You can create only a single privileged account on each ApsaraDB RDS instance. The privileged account can be used to manage all standard accounts and databases on the instance. • A privileged account allows you to manage permissions to a fine-grained level. For example, you can grant each standard account the permissions to query specific tables. • A privileged account has the permissions to disconnect all standard accounts on the instance.
Standard account	• You can create and manage standard accounts by using the ApsaraDB RDS console, API operations, or SQL statements. • You can create up to 500 standard accounts on an instance. • You must manually grant standard accounts the specific database permissions. • You cannot use a standard account to create, manage, or disconnect other accounts from databases.

Account type	Maximum number of databases	Maximum number of tables	Maximum number of accounts
Privileged account	Unlimited	< 200,000	Varies based on the engine parameter settings of the instance.
Standard account	500	< 200,000	Varies based on the engine parameter settings of the instance.

Create a privileged account

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.

4. On the **Accounts** tab, click **Create Account**.
5. On the **Create Account** page, configure the following parameters.

Parameter	Description
Database Account	Enter the name of the account. The account name must meet the following requirements: ○ The name is 1 to 16 characters in length. ○ The name starts with a lowercase letter and ends with a lowercase letter or digit. ○ The name contains lowercase letters, digits, and underscores (_).
Account Type	Select Privileged Account.
Password	Enter the password of the account. The password must meet the following requirements: ○ The password is 8 to 32 characters in length. ○ The password contains at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ○ Special characters include ! @ # $ % ^ & * ( ) _ + - =
Re-enter Password	Enter the password of the account again.
Description	Optional. Enter information about the account to facilitate subsequent management. The description can be up to 256 characters in length.

6. Click **Create**.

Reset the permissions of a privileged account

If an issue occurs on the privileged account, you can enter the password of the privileged account to reset permissions. For example, you can reset the permissions if the permissions are unexpectedly revoked.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.
4. On the **Accounts** tab, find the privileged account and click **Reset Permissions** in the **Actions** column.
5. On the **Initialize Account** page, enter the password of the privileged account and click **OK**.

Create a standard account

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.

3. In the left-side navigation pane, click **Accounts** to go to the **Accounts** page.
4. On the **Accounts** tab, click **Create Account**.
5. On the **Create Account** page, configure the following parameters.

Parameter	Description
Database Account	Enter the name of the account. The account name must meet the following requirements: ○ The name is 1 to 16 characters in length. ○ The name starts with a lowercase letter and ends with a lowercase letter or digit. ○ The name contains lowercase letters, digits, and underscores (_).
Account Type	Select Standard Account.
Authorized Databases	Select one or more databases on which you want to grant permissions to the account. You can also leave this parameter empty at this time and authorize databases after the account is created. i. Select one or more databases from the Unauthorized Databases section and click **Add** to add them to the Authorized Databases section. ii. In the Authorized Databases section, select the **Read/Write**, **Read-only**, **DDL Only**, or **DML Only** permissions on each authorized database. If you want to grant the same permissions on multiple databases to the account, click the button in the upper-right corner of the section. The button may appear as **Set All to Read/Write**.
Password	Enter the password of the account. The password must meet the following requirements: ○ The password is 8 to 32 characters in length. ○ The password contains at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters. ○ Special characters include ! @ # $ % ^ & * ( ) _ + - =
Re-enter Password	Enter the password of the account again.
Description	Optional. Enter information about the account to facilitate subsequent management. The description can be up to 256 characters in length.

6. Click **Create**.

Account permissions

Account type	Authorization type	Permission				
Privileged account	None	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	RELOAD	PROCESS	REFERENCES	INDEX
		ALTER	CREATE TEMPORARY TABLES	LOCK TABLES	EXECUTE	REPLICATION SLAVE
		REPLICATION CLIENT	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		CREATE USER	EVENT	TRIGGER	None	None
Standard account	**Read-only**	SELECT	LOCK TABLES	SHOW VIEW	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT	None	None	None	None
	Read/write	SELECT	INSERT	UPDATE	DELETE	CREATE
		DROP	REFERENCES	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	EXECUTE	CREATE VIEW	SHOW VIEW	CREATE ROUTINE
		ALTER ROUTINE	EVENT	TRIGGER	PROCESS	REPLICATION SLAVE
		REPLICATION CLIENT	None	None	None	None
	DDL-only	CREATE	DROP	INDEX	ALTER	CREATE TEMPORARY TABLES
		LOCK TABLES	CREATE VIEW	SHOW VIEW	CREATE ROUTINE	ALTER ROUTINE
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT	None	None
		SELECT	INSERT	UPDATE	DELETE	CREATE TEMPORARY TABLES

Account type	Authorization type	Permission				
		LOCK TABLES	EXECUTE	SHOW VIEW	EVENT	TRIGGER
		PROCESS	REPLICATION SLAVE	REPLICATION CLIENT	None	None

5.2. Reset the password

You can use the ApsaraDB RDS console to reset the password of your database account.

Prerequisites

The instance is in the **Running** state.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts**.
4. Find an account and click **Reset Password** in the **Actions** column.
5. In the dialog box that appears, enter and confirm the new password, and then click **OK**.

> **Note** The password must meet the following requirements:
>
> - The password is 8 to 32 characters in length.
> - The password contains at least three of the following character types: uppercase letters, lowercase letters, digits, and special characters.
> - Special characters include
>
> ! @ # $ % ^ & * ( ) _ + - =

5.3. Edit account permissions

You can edit the account permissions of your ApsaraDB RDS instance at any time.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts**.
4. Find an account and click **Edit Permissions** in the **Actions** column.

> **Note** You can edit the permissions of a standard account. The permissions of privileged accounts can only be reset to the default settings and cannot be changed to a specific set of permissions.

5. Configure the following parameters.

Parameter	Description
Authorized Databases	In the **Unauthorized Databases** section, select a database and click **Add** to authorize the database. In the **Authorized Databases** section, select a database and click **Remove** to remove the permissions from the database.
Permission	You can set permissions on each database in the Authorized Database section. You can also click the button such as **Set All to Read/Write** in the upper-right corner to set the permissions of the account on all authorized databases. ○ **Read-only**: grants the account read-only permissions on databases. ○ **Read/Write**: grants the account read and write permissions on databases. ○ **DDL Only**: grants the account DDL permissions on databases. ○ **DML Only**: grants the account DML permissions on databases.

6. Click **OK**.

5.4. Delete an account

You can delete a database account in the ApsaraDB RDS console.

Prerequisites

You can use the console to delete privileged and standard accounts that are no longer used.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Accounts**.
4. Find the account that you want to delete and click **Delete** in the Actions column.
5. In the message that appears, click **Confirm**.

> **Note** Accounts in the **Processing** state cannot be deleted.

6.Databases

6.1. Create a database

After you create an ApsaraDB RDS instance and configure its IP address whitelist, you must create a database and an account on the instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Databases**.
4. Click **Create Database**. On the page that appears, configure the following parameters.

Parameter	Description
Database Name	◦ The name must be 1 to 64 characters in length. ◦ The name must start with a letter and end with a letter or a digit. ◦ The name can contain lowercase letters, digits, underscores (_), and hyphens (-). ◦ The name must be unique within the instance.
Supported Character Sets	Select **utf8**, **gbk**, **latin1**, **utf8mb4**, or **all**. If you want to use other character sets, select **all**, and then select the required character set from the list.
Description	Optional. Enter information about the database to facilitate subsequent management. The description must be 2 to 256 characters in length.

5. Click **Create**.

6.2. Delete a database

You can delete databases that are no longer used in the ApsaraDB RDS console.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Databases**.
4. Find the database that you want to delete and click **Delete** in the **Actions** column.
5. In the Delete Database message, click **Confirm**.

7.Database connection

7.1. Change the endpoint and port number of an instance

This topic describes how to view and change the endpoint and port number of an ApsaraDB RDS instance.

View the endpoint and port number

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the **Basic Information** section, view the internal and public endpoints and port numbers.

Change the endpoint and port number of an instance

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. In the upper-right corner of the Database Connection section, click **Change Endpoint**.
5. In the dialog box that appears, set Connection Type, Endpoint, and Port, and then click **OK**.

> Note
> - The prefix of an endpoint must be 8 to 64 characters in length and can contain letters, digits, and hyphens (-). It must start with a lowercase letter.
> - The port number must be a value within the range of 1000 to 5999.

7.2. Apply for and release an internal endpoint or a public endpoint for an instance

ApsaraDB RDS supports two types of endpoints: internal endpoints and public endpoints. The default type of the endpoint used to connect to an ApsaraDB RDS instance is determined by the network connection type selected when you create the instance. This topic describes how to apply for and release an internal endpoint or a public endpoint for an ApsaraDB RDS instance.

Apply for an internal endpoint or a public endpoint

If you set **Connection Type** to **Internet** when you create an ApsaraDB RDS instance, the database system assigns a public endpoint to the instance and you can apply for an internal endpoint. Otherwise, the database system assigns an internal endpoint to the instance, and you can apply for a public endpoint.

1. Log on to the ApsaraDB for RDS console.

2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. Apply for an internal endpoint or a public endpoint:
   - To apply for a public endpoint, click **Apply for Public Endpoint**.
   - To apply for en internal endpoint, click **Apply for Internal Endpoint**.
5. In the message that appears, click **OK**.

Release an internal endpoint or a public endpoint

If an endpoint is no longer needed, you can release the endpoint to ensure instance security.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. Release an internal endpoint or a public endpoint:
   - To release a public endpoint, click **Release Public Endpoint**.
   - To release an internal endpoint, click **Release Internal Endpoint**.
5. In the message that appears, click **OK**.

FAQ

- Q: Can I change the endpoints and port numbers of my ApsaraDB RDS instance?

  A: No, you cannot change the endpoints of your ApsaraDB RDS instance. You can change the prefixes of the endpoints. You can also change the port numbers of your instance. For more information, see Change the endpoint and port number of an instance.

- Q: Can I configure the endpoints of my ApsaraDB RDS instances to static IP addresses?

  A: No, you cannot configure the endpoints of your ApsaraDB RDS instance to static IP addresses. Both primary/secondary switchovers and specification changes may cause changes to the IP addresses. We recommend that you connect to your instance by using an endpoint. This allows you to minimize impacts on your workloads and eliminates the need to modify configuration data on your application.

- Q: How do I connect to my ApsaraDB RDS instance by using a public endpoint?

  A: You can connect to your ApsaraDB RDS instance from an ECS instance or a database client. For more information, see Connect to an ApsaraDB RDS for MySQL instance.

7.3. Use DMS to log on to an ApsaraDB RDS instance

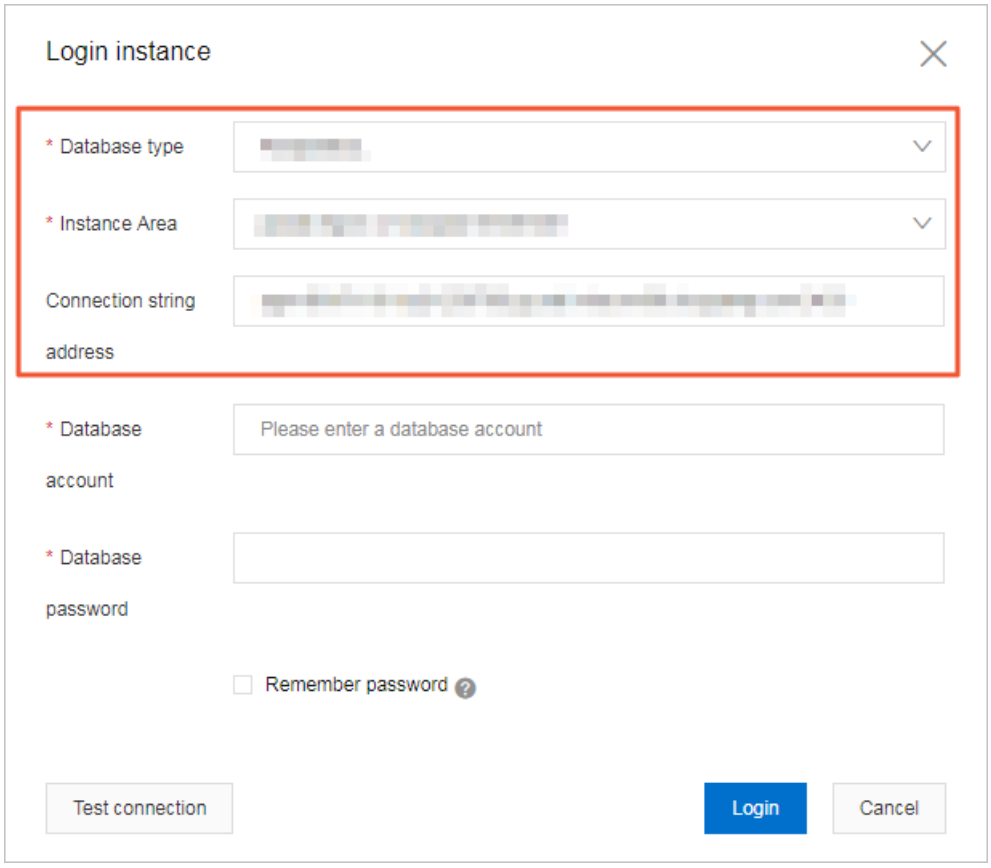
This topic describes how to use Data Management (DMS) to log on to an ApsaraDB RDS instance.

Prerequisites

An IP address whitelist is configured. For more information about how to configure an IP address whitelist, see Configure a whitelist.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. Click **Log On to DB** in the upper-right corner of the page.
4. In the **Login instance** dialog box of the **DMS** console, check values of **Database type**, **Instance Area**, and **Connection string address**. If the information is correct, enter **Database account** and **Database password**, as shown in the following figure.



Parameter	Description
Database type	The engine of the database. By default, the engine of the database to be connected is displayed.
Instance Area	The region where the instance is deployed. By default, the region of the current instance is displayed.
Connection string address	The endpoint of the instance. By default, the endpoint of the current instance is displayed.
Database account	The account of the database to be connected.
Database password	The password of the account used to connect to the database.

5. Click **Login**.

> **Note** If you want the browser to remember the password, select **Remember password** before you click **Login**.

7.4. Configure the hybrid access solution for an instance

This topic describes how to configure the hybrid access solution for an ApsaraDB RDS instance. This solution allows you to retain both the classic network endpoint and Virtual Private Cloud (VPC) endpoint of your ApsaraDB RDS instance. This way, you can migrate your ApsaraDB RDS instance from the classic network to a VPC without network interruptions.

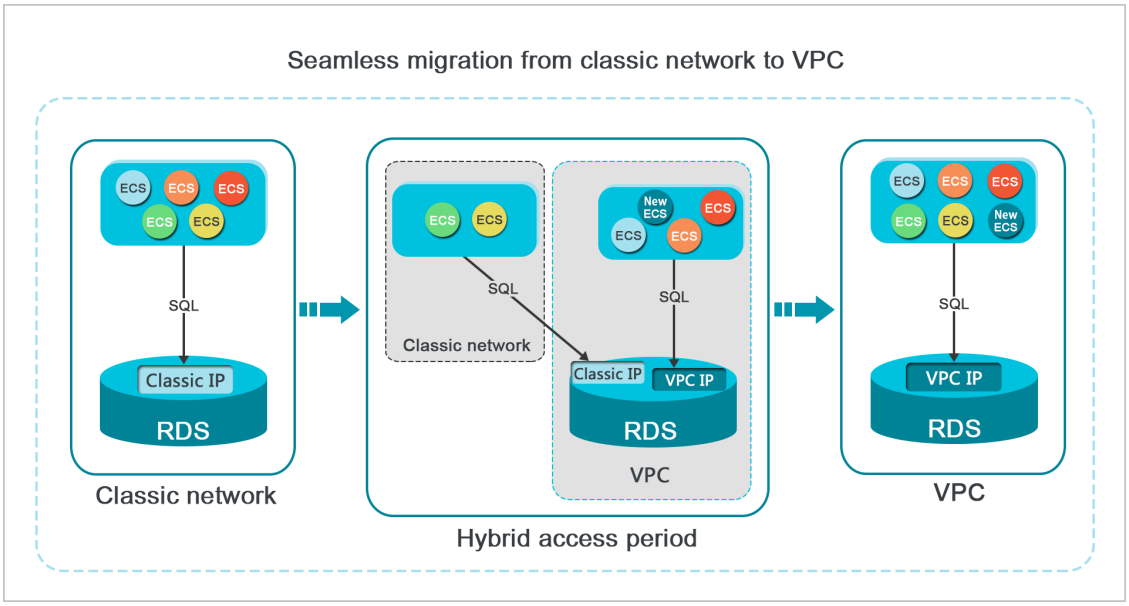
Background information

When you migrate your ApsaraDB RDS instance from the classic network to a VPC, the internal classic network endpoint of the instance changes to the internal VPC endpoint. In this case, the endpoint remains unchanged, but the IP address that is bound to the endpoint changes. This change causes a network interruption of 30 seconds or less, and Elastic Compute Service (ECS) instances located in the classic network can no longer connect to your ApsaraDB RDS instance over an internal network. To facilitate a smooth migration, ApsaraDB RDS provides the hybrid access solution.

Hybrid access refers to the ability of your ApsaraDB RDS instance to be connected by both ECS instances located in the classic network and ECS instances located in a VPC. During the validity period of the hybrid access solution, ApsaraDB RDS retains the internal classic network endpoint and generates an internal VPC endpoint. This prevents network interruptions when you migrate your instance from the classic network to a VPC.

For security and performance purposes, we recommend that you use only the internal VPC endpoint. You must specify a validity period for the hybrid access solution. When the validity period expires, ApsaraDB RDS releases the internal classic network endpoint and applications are unable to use the endpoint to connect to your instance. Therefore, you must add the internal VPC endpoint to your applications before the validity period expires. This ensures a smooth migration and prevents interruptions to your workloads.

For example, assume that a company uses the hybrid access solution to migrate its ApsaraDB RDS instance from the classic network to a VPC. During the validity period of the hybrid access solution, some applications use the internal VPC endpoint to connect to the ApsaraDB RDS instance, whereas the others continue to use the internal classic network endpoint to connect to the instance. When all applications of the company can use the internal VPC endpoint to connect to the instance, the internal classic network endpoint can be released. The following figure demonstrates the scenario.



Limits

During the validity period of the hybrid access solution, your ApsaraDB RDS instance has the following limits:

- The network type of the instance cannot be changed to classic network.
- The instance cannot be migrated to another zone.

Prerequisites

- The ApsaraDB RDS instance resides in the classic network.
- Available VPCs and vSwitches exist in the zone where the ApsaraDB RDS instance resides.
- Your ApsaraDB RDS instance provides an internal endpoint. If no internal endpoint exists, you must apply for one. For more information, see Apply for and release an internal endpoint or a public endpoint for an instance.

Change the network type from classic network to VPC

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. On the Instance Connection tab, click **Switch to VPC**.
5. In the Switch to VPC dialog box, select a VPC and a vSwitch and specify whether to retain the classic network endpoint.

   Clear or select the **Reserve Original Classic Network Endpoint** check box based on the details described in the following table.

Action	Description
Clear the Reserve Original Classic Network Endpoint check box	The classic network endpoint is not retained and changes to a VPC endpoint. When you change the network type from classic network to VPC, a network interruption of 30 seconds occurs. When this occurs, ECS instances located in the classic network are disconnected from your ApsaraDB RDS instance.
Select the Reserve Original Classic Network Endpoint check box	The classic network endpoint is retained, and a new VPC endpoint is generated. In this case, your ApsaraDB RDS instance runs in hybrid access mode. Both ECS instances located in the classic network and ECS instances located in the selected VPC can access your ApsaraDB RDS instance over an internal network. When you change the network type from classic network to VPC, no network interruptions occur. ECS instances located in the classic network are still connected with your ApsaraDB RDS instance until the classic network endpoint expires. Specify the expiration date of the classic network endpoint. Before the classic network endpoint expires, you must add the new VPC endpoint to your applications that run on the ECS instances located in the selected VPC. This allows ApsaraDB RDS to migrate your workloads to the selected VPC without network interruptions.

6. Add the internal IP addresses of ECS instances located in the selected VPC to an IP address whitelist of the VPC network type. This allows the ECS instances to connect to your ApsaraDB RDS instance over an internal network. If no IP address whitelists of the VPC network type are available, create one. For more information, see Configure a whitelist.

Change the expiration date of the internal classic network endpoint

During the validity period of the hybrid access solution, you can change the expiration date of the classic network endpoint based on your business requirements. The expiration date is immediately recalculated starting from the day when you make the change. For example, assume that the classic network endpoint is configured to expire on August 18, 2017. On August 15, 2017, you increase the validity period of the classic network endpoint by 14 days. In this case, ApsaraDB RDS releases the classic network endpoint on August 29, 2017.

To change the expiration date, perform the following operations:

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. Click **Change Expiration Time**.
5. In the **Change Expiration Time** dialog box, select an expiration date and click **OK**.

7.5. Change the network type of an instance

This topic describes how to change the network type of an ApsaraDB RDS instance between classic network and Virtual Private Cloud (VPC).

Context

- Classic network: ApsaraDB RDS instances in the classic network are not isolated. Unauthorized access to these instances can be blocked only by IP address whitelists.
- VPC: Each VPC is an isolated virtual network. We recommend that you select the VPC type because it is more secure than the classic network.

  You can configure route tables, CIDR blocks, and gateways in a VPC. To smoothly migrate applications to the cloud, you can use the leased line or VPN method to create a virtual data center that consists of your data center and a VPC.

Change the network type from VPC to classic network

Precautions

- After you change the network type from VPC to classic network, the internal endpoint of your ApsaraDB RDS instance remains unchanged. However, the IP address that is associated with the internal endpoint changes.
- After you change the network type from VPC to classic network, Elastic Compute Service (ECS) instances located in the same VPC as your ApsaraDB RDS instance can no longer connect to your ApsaraDB RDS instance by using the internal endpoint. You must update the endpoint for the applications deployed on the ECS instances.
- When you change the network type, a network interruption of 30 seconds may occur. To avoid business interruption, change the network type during off-peak hours or make sure that your application is configured to automatically reconnect to the instance.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. In the upper-right corner of the Database Connection section, click **Switch to Classic Network**.
5. In the message that appears, click **OK**.

Change the network type from classic network to VPC

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. In the upper-right corner of the Database Connection section, click **Switch to VPC**.
5. In the Switch to VPC dialog box, select a VPC and a vSwitch, and then select or clear **Reserve Original Classic Network Endpoint**. Click OK. For more information about **Reserve Original Classic Network Endpoint**, see Hybrid access from both the classic network and VPCs.

7.6. Change the VPC and vSwitch for an instance

This topic describes how to change the virtual private cloud (VPC) and vSwitch for an ApsaraDB RDS instance.

Prerequisites

The instance is deployed in a VPC.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Connection**.
4. In the upper-right corner of the Database Connection section, click **Switch vSwitch**.
5. Select a VPC and a vSwitch, and then click **OK**.
6. In the message that appears, click **OK**.

> **Note**
>
> - A network interruption of 30 seconds may occur when you switch the VPC and vSwitch of an ApsaraDB RDS instance. Make sure that your application is configured to automatically reconnect to the instance.
> - We recommend that you clear the cache immediately after the instance is switched to a new VPC and vSwitch. Otherwise, data can be read but not written.

8.Database proxy

8.1. Configure dedicated proxy

This topic describes the dedicated proxy feature provided by ApsaraDB RDS. The dedicated proxy feature provides advanced features such as read/write splitting, connection pooling, and transaction splitting.

Context

The dedicated proxy feature uses dedicated proxy computing resources. This feature has the following benefits:

- A unified proxy endpoint is provided to connect to all the dedicated proxies that are enabled on your ApsaraDB RDS instance. This reduces maintenance costs by eliminating the need to update the endpoints on your application. The proxy endpoint remains valid until you release the dedicated proxies. For example, you can enable read/write splitting during peak hours and then disable read/write splitting and release read-only instances after peak hours end. In these cases, you do not need to update the endpoints on your application because the proxy endpoint remains connected.
- Dedicated proxies serve your ApsaraDB RDS instance and its read-only instances exclusively. You do not need to compete with other users for resources. This ensures service stability.
- Dedicated proxies are scalable. You can add dedicated proxies based on your business requirements to handle more workloads.

Limits

- Dedicated proxies do not support SSL encryption.
- Dedicated proxies do not support compression protocols.

Precautions

- When you change the specifications of your ApsaraDB RDS instance or its read-only instances, a service interruption may occur.
- If you connect your application to the proxy endpoint, all requests that are encapsulated in transactions are routed to your ApsaraDB RDS instance. This applies when the transaction splitting feature is not enabled.
- If a proxy endpoint is used to implement read/write splitting, read consistency cannot be ensured for the requests that are not encapsulated in transactions. If you require read consistency for these requests, you can encapsulate these requests in transactions.
- If a proxy endpoint is used for connection, the `SHOW PROCESSLIST` statement returns a result set for each query. The result set consists of the query results from the primary and read-only instances.
- If you execute multi-statements or stored procedures, the read/write splitting feature is disabled and all subsequent requests over the current connection are routed to the primary ApsaraDB RDS instance. To enable the read/write splitting feature again, you must close the current connection and establish a new connection.
- The dedicated proxy feature supports the `/*FORCE_MASTER*/` and `/*FORCE_SLAVE*/` hints. However, requests that contain hints have the highest route priorities and are not constrained by consistency or transaction limits. Before you use these hints, you must check whether these hints are suitable for your workloads. A hint cannot contain statements that change environment variables. An

example is `/*FORCE_SLAVE*/ set names utf8;`. Otherwise, an error may occur in the subsequent procedure.

- After you enable the dedicated proxy feature, each connection is replicated to the primary ApsaraDB RDS instance and all of its read-only instances in compliance with the 1:N connection model. We recommend that you specify the same connection specifications for these instances. If these instances have different connection specifications, the number of connections allowed depends on the lowest connection specifications among these instances.
- If you create or restart a read-only instance after you enable the dedicated proxy feature, only the requests sent over new connections are routed to the new or restarted read-only instance.
- The **max_prepared_stmt_count** parameter must be set to the same value for the primary ApsaraDB RDS instance and all of its read-only instances.

Enable the dedicated proxy feature

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. Click **Enable now**.

> **Note**
> - If the network type of the instance is **Classic Network**, the dedicated proxy feature is enabled.
> - If the network type of the instance is **VPC**, you must select a vSwitch and click **Enable** in the dialog box that appears.

Overview of the Proxy Service tab

When the dedicated proxy feature is enabled, you can use the generated proxy endpoint to implement features such as read/write splitting, short-lived connection optimization, and transaction splitting.

Section	Parameter	Description
	Instance ID	The ID of the primary ApsaraDB RDS instance.
	Enabled Proxies	The number of enabled dedicated proxies. You can enable more dedicated proxies to increase the maximum number of requests that can be processed.
	Read/Write Splitting	Specifies whether to enable the read/write splitting feature for the proxy endpoint. For more information, see Read/write splitting.

Section	Parameter	Description
Proxy Endpoint	**Short-Lived Connection Optimization**	The type of connection pool for the proxy endpoint. This feature is suitable for scenarios where PHP short-lived connections are established. For more information, see Short-lived connection optimization. **Note** You can click **Enable** or **Disable** to the right of Short-Lived Connection Optimization to enable or disable this feature.
	Transaction Splitting	Specifies whether to enable the transaction splitting feature for the proxy endpoint. For information, see Transaction splitting. **Note** You can click **Enable** or **Disable** to the right of Transaction Splitting to enable or disable this feature.
	Endpoint	The proxy endpoint that is generated when the dedicated proxy feature is enabled. This endpoint connects to all the dedicated proxies that are enabled on the ApsaraDB RDS instance. The read/write splitting feature is also bound to this endpoint. **Note** You can click **Copy Address** to the right of Endpoint to copy the endpoint.
	Port	The port number that is used to connect to the proxy endpoint.
	Endpoint Type	The network type of the proxy endpoint.
Proxy	**Proxy Type**	The type of proxy that is enabled. Only **Dedicated Proxy** is supported.
	CPU and Memory	The CPU and memory specifications of the dedicated proxies. Only 2 Cores, 4 GB is supported.

Section	Parameter	Description
	Enabled Proxies	The number of dedicated proxies that are enabled on the primary ApsaraDB RDS instance. Up to 60 dedicated proxies are supported. **Note** We recommend that you use the following formula to determine the number of dedicated proxies to specify: (Total number of CPU cores of your ApsaraDB RDS instance and its read-only instances)/8, rounded up to the nearest integer. For example, if your ApsaraDB RDS instance has 8 CPU cores and its read-only instances have 4 CPU cores, the recommended number of dedicated proxies is 2, as calculated in the following formula: ⌈(8 + 4)/8⌉ = 2.

Adjust the number of dedicated proxies

Note When you adjust the number of dedicated proxies, a service interruption may occur. Make sure that your application is configured to automatically reconnect to the instance.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. In the **Proxy** section of the Proxy Service tab, change the number in the **Adjusted Proxies** column and click **Apply** in the **Adjustment Plan** column.
5. In the Configure Proxy Resources dialog box, select **Migrate Immediately** to apply the change. You can also select **Next Maintenance Period** to set a maintenance window for the change to take effect. Click **OK**.

Disable the dedicated proxy feature

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. In the upper-right corner of the page, click **Disable Proxy Service**.
5. Click **OK**.

8.2. Configure short-lived connection optimization

This topic describes the short-lived connection optimization feature provided by ApsaraDB RDS in its dedicated proxy feature.

Prerequisites

The database proxy feature is enabled for your ApsaraDB RDS instance. For more information, see Dedicated proxy.

Context

The short-lived connection optimization feature is used to reduce workloads on the ApsaraDB RDS instance caused by frequent short-lived connections. When a client is disconnected, the system checks whether the closed connection is idle. If the connection is considered idle, the dedicated proxy retains the connection in the connection pool for a short period of time. When the client initiates a request for access to your instance again, the dedicated proxy searches the connection pool for an idle connection that matches the request. The connection pool is matched based on the values of the user, clientip, and dbname fields in the request. If the dedicated proxy finds an idle connection that matches the request, it reuses the matched idle connection. If no idle connection can be matched, a new connection is established with your instance to reduce database connection overheads.

> **Note** The short-lived connection optimization feature does not reduce the number of concurrent connections with the instance. It decreases the frequency at which connections are established between an application and your instance to reduce overheads of the primary MySQL thread and improve efficiency to process business requests. However, idle connections in the connection pool still occupy the database threads for a short period of time.

Precautions

You cannot configure different permissions for different source IP addresses by using the same account. Otherwise, errors may occur when connections in the connection pool are reused. For example, if a user account has permissions on database_a when its source IP address is 192.168.1.1 but does not have permissions on database_a when its source IP address is 192.168.1.2, the short-lived connection optimization feature may encounter permission errors.

Enable short-lived connection optimization

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. On the **Proxy Service** tab, click **Enable** to the right of **Short-Lived Connection Optimization**.

8.3. Configure transaction splitting

This topic describes the transaction splitting feature provided by the dedicated proxy of ApsaraDB RDS. This feature identifies and distributes read requests initiated before write requests within a transaction to read-only instances. This reduces workloads on the primary instance.

Prerequisites

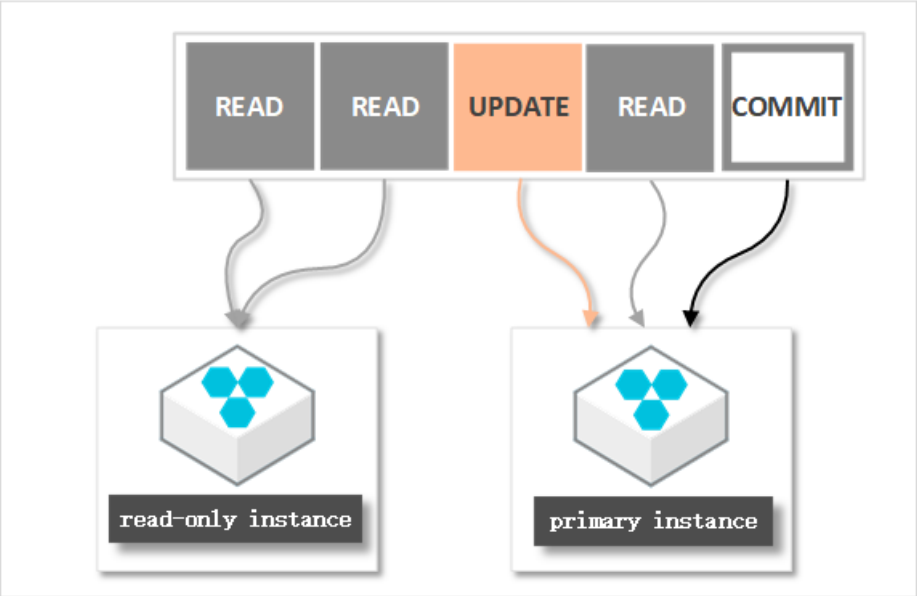
The dedicated proxy feature is enabled for your ApsaraDB RDS instance. For more information, see Dedicated proxy.

Context

By default, the dedicated proxy sends all requests in transactions to the primary instance to ensure the correctness of the transactions. If the framework encapsulates all requests in transactions, the primary instance becomes heavily loaded. In this case, you can enable the transaction splitting feature.

When transaction splitting is enabled and the default isolation level READ COMMITTED is used, the ApsaraDB RDS instance starts a transaction only for write requests when autocommit is disabled (set autocommit=0). Read requests that arrive before the transaction is started are distributed to read-only instances by the load balancer.

> Note
>
> - Explicit transactions do not support splitting, such as transactions started by using the BEGIN or START statement.
> - After you enable the transaction splitting feature, global consistency cannot be ensured. Before you enable this feature, we recommend that you evaluate whether this feature is suitable for your workloads.



Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. On the **Proxy Service** tab, click **Enable** to the right of **Transaction Splitting**.

> Note
>
> - When you no longer need transaction splitting, you can click **Disable** to the right of **Transaction Splitting**.
> - The operation to enable or disable transaction splitting takes effect only on new connections.

8.4. Read/write splitting

8.4.1. Enable read/write splitting

This topic describes the read/write splitting feature. This feature allows ApsaraDB RDS to route read and write requests to the primary and read-only instances based on the dedicated proxy endpoint (also called read/write splitting endpoint).

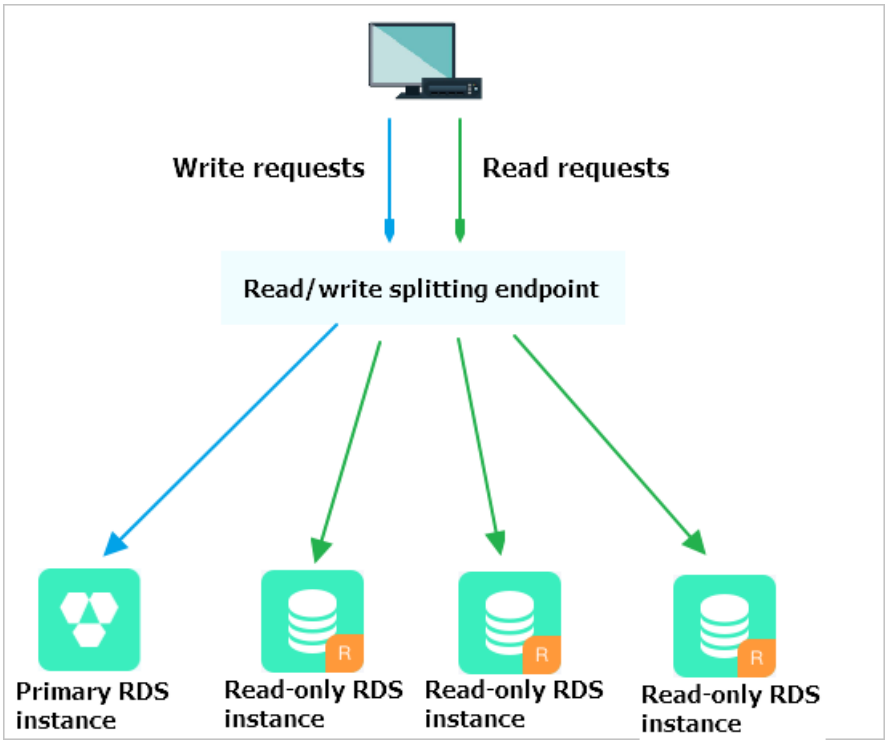
Prerequisites

- The database proxy or dedicated proxy feature is enabled. For more information, see Enable the dedicated proxy feature or Enable the dedicated proxy feature.
- At least one read-only instance is created. For more information about how to create a read-only instance, see Create a read-only instance.

Context

If your database system receives a large number of read requests and a small number of write requests, a single primary ApsaraDB RDS instance may fail to process read requests and your workloads may be interrupted. In this case, you can create one or more read-only ApsaraDB RDS instances to offload read requests from the primary instance and increase the read capability of your database system. For more information, see Create a read-only instance.

After you create read-only instances, you can enable read/write splitting. In this case, a read/write splitting endpoint is provided. After you add the endpoint to your application, write requests are routed to the primary instance and read requests are routed to the read-only instances.



Differences between the read/write splitting endpoint and the internal and public endpoints

After you enable read/write splitting and add the read/write splitting endpoint to your application, all requests are first routed to this endpoint, and then to the primary and read-only instances based on the request types and the read weights of these instances.

If the internal or public endpoint of the primary instance is added to your application, all requests are routed to the primary instance. To implement read/write splitting, you must add the endpoints and read weights of the primary and read-only instances to your application.

Logic to route requests

- The following requests are routed only to the primary instance:
  - All requests that are used to execute DML statements such as INSERT, UPDATE, DELETE, and SELECT FOR UPDATE.
  - All requests that are used to execute DDL statements, such as the DDL statements that are used to create databases or tables, delete databases or tables, and change schemas or permissions.
  - All requests that are encapsulated in transactions.
  - Requests that are used to call user-defined functions.
  - Requests that are used to run stored procedures.
  - Requests that are used to execute EXECUTE statements.
  - Requests that are used to run multi-statement queries. For more information, see Multi-statement.
  - Requests that involve temporary tables.
  - Requests that are used to execute SELECT last_insert_id() statements.
  - All requests that are used to query or reconfigure user variables.
  - Requests that are used to execute SHOW PROCESSLIST statements.
  - Requests that are used to execute KILL statements in SQL. These statements are different from the KILL commands in Linux.
- The following requests are routed to the primary instance or its read-only instances:
  - Read requests that are not encapsulated in transactions.
  - Requests that are used to execute COM_STMT_EXECUTE statements.
- The following requests are routed to all the instances:
  - All requests that are used to modify system variables.
  - Requests that are used to execute USE statements.
  - Requests that are used to execute COM_STMT_PREPARE statements.
  - Requests that are used to execute COM_CHANGE_USER, COM_QUIT, and COM_SET_OPTION statements.

Benefits

- Easier maintenance by using a unified endpoint

  If you do not enable the read/write splitting feature, you must add the endpoints of the primary and read-only instances to your application. This way, your database system routes write requests to the primary instance and read requests to the read-only instances.

  If you enable the read/write splitting feature, you can use a dedicated proxy endpoint to implement read/write splitting. After your application is connected to this endpoint, your database system routes read and write requests to the primary and read-only instances based on the read weights of these instances. This reduces maintenance costs.

  You can also create read-only instances to improve the read capability of your database system. You do not need to modify the configuration data on your application.

- Higher performance and lower maintenance costs by using a native link

  You can build your own proxy layer on the cloud to implement read/write splitting. In this case, data needs to be parsed and forwarded by multiple components before the data reaches your database system. As a result, response latencies increase. The read/write splitting feature is embedded in the ApsaraDB RDS ecosystem to reduce response latencies, increase processing speeds, and reduce maintenance costs.

- Ideal in a variety of use scenarios based on configurable read weights and thresholds

  You can specify the read weights of the primary and read-only instances. You can also specify the latency threshold for data replication to the read-only instances.

- High availability based on instance-level health checks

  The read/write splitting feature enables ApsaraDB RDS to actively check the health status of the primary and read-only instances. If a read-only instance unexpectedly breaks down or its data replication latency exceeds the specified threshold, ApsaraDB RDS stops routing read requests to the instance. ApsaraDB RDS redirects read requests that are destined for the faulty read-only instance to healthy instances in your database system. This ensures service availability in the event of faults on individual read-only instances. After the faulty read-only instance is recovered, ApsaraDB RDS resumes routing read requests to the instance.

  > **Note** To avoid single points of failure (SPOFs), we recommend that you create at least two read-only instances.

Precautions

- When you change the specifications of your ApsaraDB RDS instance or its read-only instances, a service interruption may occur.
- After you create a read-only instance, only the requests over new connections can be routed to the read-only instance.
- The dedicated proxy endpoint does not support SSL encryption.
- The dedicated proxy endpoint does not support compression.
- If a dedicated proxy endpoint is used to connect to your database system, all the requests that are encapsulated in transactions are routed to the primary instance.
- If a dedicated proxy endpoint is used to implement read/write splitting, the read consistency of the requests that are not encapsulated in transactions cannot be ensured. If you require read consistency for these requests, you can encapsulate these requests in transactions.
- If a dedicated proxy endpoint is used for connection, the `SHOW PROCESSLIST` statement returns a result set for each query. The result set consists of the query results from the primary and read-only instances.
- If the short-lived connection optimization feature is enabled, the `SHOW PROCESSLIST` statement may return idle connections.
- If you execute multi-statements or stored procedures, the read/write splitting feature is disabled and all subsequent requests over the current connection are routed to the primary ApsaraDB RDS instance. To enable the read/write splitting feature again, you must close the current connection and establish a new connection.
- The dedicated proxy feature supports the `/*FORCE_MASTER*/` and `/*FORCE_SLAVE*/` hints. However, requests that contain hints have the highest route priorities and are not constrained by consistency or transaction limits. Before you use these hints, you must check whether these hints are

suitable for your workloads. A hint cannot contain statements that change environment variables. An example is `/*FORCE_SLAVE*/ set names utf8;`. Otherwise, an error may occur in the subsequent procedure.

Prerequisite

A read-only instance is created for the primary instance. For more information, see Create a read-only instance.

Enable read/write splitting

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. On the **Read/Write Splitting** tab, click **Enable now**.
5. Configure the parameters described in the following table.

Parameter	Description
Latency Threshold	The maximum latency that is allowed for data replication from the primary instance to its read-only instances. If the latency of data replication to a read-only instance exceeds the specified threshold, ApsaraDB RDS stops routing read requests to the instance. This applies even if the instance has a high read weight. Valid values: 0 to 7200. Unit: seconds. The read-only instances may replicate data from the primary instance at a specific latency due to SQL statement execution limits. We recommend that you set this parameter to a value that is greater than or equal to 30.
Read Weight Distribution	The read weight of each instance in your database system. A higher read weight indicates more read requests to process. For example, assume that your primary instance has three attached read-only instances, and the read weights of the primary and read-only instances are 0, 100, 200, and 200. In this case, your primary instance processes only write requests, and the three read-only instances process all of the read requests at a ratio of 1:2:2. ◦ **Automatic Distribution**: Your database system assigns a read weight to each instance based on the instance specifications. After you create a read-only instance, your database system assigns a read weight to the read-only instance and adds the read-only instance to the read/write splitting link. ◦ **Customized Distribution**: You must manually specify the read weight of each instance. Valid values: 0 to 10000. After you create a read-only instance, ApsaraDB RDS sets the read weight of the read-only instance to 0. You must manually modify the read weight of the created read-only instance.

6. Click **OK**.

8.4.2. Configure read/write splitting

This topic describes how to configure the latency threshold and specify read weights for an ApsaraDB RDS instance in the ApsaraDB RDS console.

Prerequisites

Read/write splitting is enabled. For more information, see Enable read/write splitting.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Database Proxy**.
4. On the **Read/Write Splitting** tab, click **Configure Read/Write Splitting**.
5. Configure the parameters described in the following table.

Parameter	Description
Latency Threshold	The maximum latency that is allowed for data replication from the primary instance to its read-only instances. If the latency of data replication to a read-only instance exceeds the specified threshold, ApsaraDB RDS stops routing read requests to the instance. This applies even if the instance has a high read weight. Valid values: 0 to 7200. Unit: seconds. The read-only instances may replicate data from the primary instance at a specific latency due to SQL statement execution limits. We recommend that you set this parameter to a value that is greater than or equal to 30.
Read Weight Distribution	The read weight of each instance in your database system. A higher read weight indicates more read requests to process. For example, assume that your primary instance has three attached read-only instances, and the read weights of the primary and read-only instances are 0, 100, 200, and 200. In this case, your primary instance processes only write requests, and the three read-only instances process all of the read requests at a ratio of 1:2:2. ◦ **Automatic Distribution**: Your database system assigns a read weight to each instance based on the instance specifications. After you create a read-only instance, your database system assigns a read weight to the read-only instance and adds the read-only instance to the read/write splitting link. ◦ **Customized Distribution**: You must manually specify the read weight of each instance. Valid values: 0 to 10000. After you create a read-only instance, ApsaraDB RDS sets the read weight of the read-only instance to 0. You must manually modify the read weight of the created read-only instance.

6. Click **OK**.

8.4.3. Disable read/write splitting

This topic describes how to disable the read/write splitting feature of an ApsaraDB RDS instance in the ApsaraDB RDS console.

Prerequisites

Read/write splitting is enabled. For more information, see Enable read/write splitting.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.

3. In the left-side navigation pane, click **Database Proxy**.
4. On the **Read/Write Splitting** tab, click **Disable Read/Write Splitting**.
5. In the message that appears, click **Confirm**.

9.Monitoring and alerts

9.1. View resource and engine monitoring data

The ApsaraDB RDS console provides a variety of performance metrics to monitor the status of your instances.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Monitoring and Alerts**.
4. On the **Monitoring and Alerts** page, select **Resource Monitoring** or **Engine Monitoring**, and select a time range to view the corresponding monitoring data. The following table describes the metrics.

Monitoring type	Metric	Description
Resource Monitoring	**Disk Space (MB)**	The disk space usage of the instance. It consists of the following items: ◦ Instance size ◦ Data usage ◦ Log size ◦ Temporary file size ◦ Other system file size Unit: MB.
	IOPS (Input/Output Operations per Second)	The number of input/output operations per second (IOPS) of the instance.
	Total Connections	The number of active connections to the instance and the total number of connections to the instance.
	CPU Utilization and Memory Usage (%)	The CPU utilization and memory usage of the instance. These metrics do not include the CPU utilization and memory usage for the operating system.
	Network Traffic (KB)	The inbound and outbound traffic of the instance per second. Unit: KB.
	Transactions per Second (TPS)/Queries per Second (QPS)	The average number of transactions per second (TPS) and the average number of SQL statements executed per second.

Monitoring type	Metric	Description
Engine Monitoring	**InnoDB Buffer Pool Read Hit Ratio, Usage Ratio, and Dirty Block Ratio (%)**	The read hit ratio, usage ratio, and dirty block ratio of the InnoDB buffer pool.
	InnoDB Read/Write Volume (KB)	The amount of data that InnoDB reads and writes per second. Unit: KB.
	InnoDB Buffer Pool Read/Write Frequency	The number of read and write operations that InnoDB performs per second.
	InnoDB Log Read/Write/fsync	The average frequency of physical writes to log files per second by InnoDB, the frequency of log write requests, and the average frequency of fsync writes to log files.
	Temporary Tables Automatically Created on Hard Disk when MySQL Statements Are Executed	The number of temporary tables that are automatically created on the hard disk when the database executes SQL statements.
	MySQL_COMDML	The number of SQL statements that the database executes per second. The following SQL statements are included: ◦ Insert ◦ Delete ◦ Insert_Select ◦ Replace ◦ Replace_Select ◦ Select ◦ Update
	MySQL_RowDML	The numbers of operations that InnoDB performs per second. The following items are included: ◦ The number of physical writes to log files per second ◦ The number of rows that are read, updated, deleted, and inserted from InnoDB tables per second
	MyISAM Read/Write Frequency	The numbers of operations that MyISAM performs per second. The following items are included: ◦ The number of MyISAM reads and writes from the buffer pool per second ◦ The number of MyISAM reads and writes from the hard disk per second

Monitoring type	Metric	Description
	MyISAM Key Buffer Read/Write/Usage Ratio (%)	The read hit ratio, write hit ratio, and usage of the MyISAM key buffer per second.

9.2. Set a monitoring frequency

This topic describes how to set a monitoring frequency for an ApsaraDB RDS instance.

Context

ApsaraDB RDS provides the following monitoring frequencies:

- Every 5 seconds for the first seven days. After the seven days, performance metrics are monitored every 60 seconds.
- Every 60 seconds.
- Every 300 seconds.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Monitoring and Alerts**.
4. On the Monitoring tab, click **Set Monitoring Frequency**.
5. In the **Set Monitoring Frequency** dialog box, select a new monitoring frequency.

   > **Note** If an ApsaraDB RDS instance runs the RDS Basic Edition or its memory capacity is less than 8 GB, the **Every 5 Seconds** monitoring frequency is not supported.

6. Click **OK**.

10.Data security

10.1. Configure an IP address whitelist for an ApsaraDB RDS instance

After you create an ApsaraDB RDS instance, you must add the IP addresses or CIDR blocks that are used for database access to the IP address whitelist of the instance to ensure database security and reliability.

Context

IP address whitelists make your ApsaraDB RDS instance more secure and do not interrupt the operations of your ApsaraDB RDS instance when you configure whitelists. We recommend that you maintain your IP address whitelists on a regular basis.

To configure a whitelist, you can perform the following operations:

- Configure an IP address whitelist: Add IP addresses to allow them to connect to the ApsaraDB RDS instance.
- Configure an Elastic Compute Service (ECS) security group: Add an ECS security group for the ApsaraDB RDS instance to allow ECS instances in the group to connect to the ApsaraDB RDS instance.

Precautions

- The default IP address whitelist can be modified or cleared, but cannot be deleted.
- You can add up to 1,000 IP addresses or CIDR blocks to a whitelist. If you want to add a large number of IP addresses, we recommend that you merge them into CIDR blocks, such as 192.168.1.0/24.

Introduction to IPv6

IPv4 addresses are widely used, but the limited number of IPv4 addresses restricts the development of the Internet. Compared with IPv4 addresses, IPv6 addresses are more sufficient and allow more types of devices to access the Internet. ApsaraDB RDS supports both IPv4 and IPv6 addresses.

The following table describes the differences between IPv4 and IPv6.

Item	IPv4	IPv6
Address length	32 bits (4 bytes)	128 bits (16 bytes)
Number of addresses	2^32	2^128

Item	IPv4	IPv6
Address format	xxx.xxx.xxx.xxx Where xxx is a decimal number that can range from 0 to 255. Each x is a decimal integer, and leading zeros can be omitted. Example: 192.168.1.1	xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx Where each x is a hexadecimal number, and leading zeros can be omitted. You can use a double colon (::) once in an IPv6 address to indicate a series of zeros. Example: CDDC:0000:0000:0000:8475:1111:3900:2020
Address Resolution Protocol (ARP)	Uses broadcast ARP Request frames to resolve an IP address to a link layer address.	Uses multicast neighbor solicitation messages to resolve an IP address to a link layer address.
Security	Implements a security mechanism based on applications and cannot provide protections at the IP layer.	Supports packet fragmentation to ensure data confidentiality and integrity and provides security at the IP layer.
LAN connection	Connects to LANs by using network interfaces.	Can work with Ethernet adapters and is supported over virtual Ethernet networks between logical partitions.
Address type	• Unicast address • Multicast address • Broadcast address	• Unicast address • Multicast address • Anycast address

Create an IP address whitelist

Each IP address whitelist of an ApsaraDB RDS instance can contain **IPv4** or **IPv6** addresses. By default, the system provides an IP address whitelist of the **IPv4** type. If you want an IP address whitelist of the **IPv6** type, manually create one.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. On the **Whitelist Settings** tab, click **Create Whitelist**. In the dialog box that appears, configure the following parameters.

Parameter	Description

Parameter	Description
Whitelist Name	The name of the IP address whitelist. **Note** ◦ The name can contain lowercase letters, digits, and underscores (_). ◦ The name must start with a lowercase letter and end with a lowercase letter or digit. ◦ The name must be 2 to 32 characters in length.
IP Type	The IP type of the IP address whitelist. Valid values: ◦ **IPv4** ◦ **IPv6** **Note** For more information about the differences between IPv4 and IPv6, see the "Introduction to IPv6" section of this topic.
IP Addresses	The IP addresses that are allowed to access the instance.

Configure an IP address whitelist

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. On the **Whitelist Settings** tab, click **Edit** corresponding to an IP address whitelist.

   Note If you want to connect an ECS instance to an ApsaraDB 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.

5. In the **Edit Whitelist** dialog box, enter the IP addresses or CIDR blocks that are allowed to access the instance and click **OK**.

Note

- Limits for **IPv4** addresses:
  - You must separate multiple IP addresses with commas (,). A maximum of 1,000 different IP addresses can be added.

    Supported formats are `0.0.0.0/0`, IP addresses such as `10.23.12.24`, or CIDR blocks such as `10.23.12.24/24`. `/24` indicates the length of the IP address prefix in the CIDR block. An IP address prefix can contain 1 to 32 bits.
  - If an IP address whitelist is empty or contains `0.0.0.0/0`, all IP addresses can access the ApsaraDB RDS instance. This may cause security risks to the instance. Proceed with caution.
- Limits for **IPv6** addresses:
  - You must separate multiple IP addresses with commas (,). A maximum of 1,000 different IP addresses can be added.

    Supported formats are `::`, IP addresses such as `0:0:0:0:0:0:0:1`, or CIDR blocks such as `0:0:0:0:0:0:0:1/24`. `/24` indicates the length of the IP address prefix in the CIDR block. An IP address prefix can contain 1 to 128 bits.
  - If an IP address whitelist is empty or contains only `::`, all IP addresses can access the ApsaraDB RDS instance. This may cause security risks to the instance. Proceed with caution.
- You cannot specify both **IPv4** and **IPv6** addresses in a single IP address whitelist. If you want to specify both IPv4 and IPv6 addresses, specify them in separate IP address whitelists.
- If you click **Add Internal IP Addresses of ECS Instances**, the IP addresses of all the ECS instances that are created within your Apsara Stack tenant account appear. Then, you can select the IP addresses and add them to an IP address whitelist.

10.2. Configure SSL encryption

This topic describes how to enhance endpoint security. You can enable Secure Sockets Layer (SSL) encryption and install SSL certificates that are issued by certificate authorities (CAs) to the required application services. SSL is used at the transport layer to encrypt network connections and enhance the security and integrity of communication data. However, SSL increases the response time.

Prerequisites

Your ApsaraDB RDS instance runs one of the following MySQL versions and RDS editions:

- MySQL 8.0 on RDS High-availability Edition (with local SSDs)
- MySQL 5.7 on RDS High-availability Edition (with local SSDs)
- MySQL 5.6 on RDS High-availability Edition (with local SSDs)

Precautions

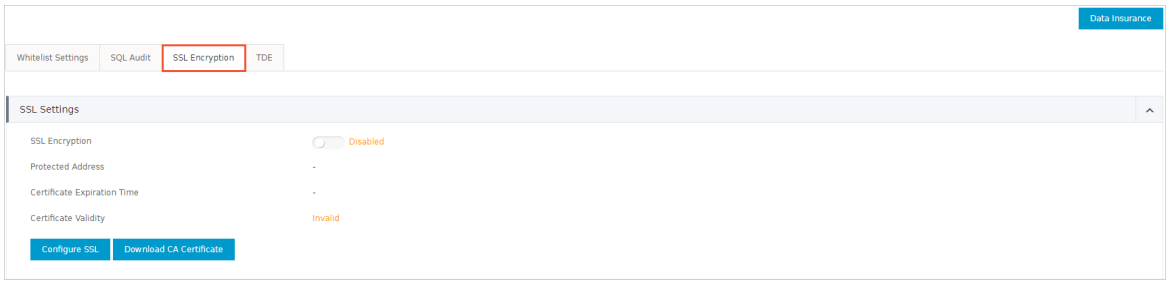
- An SSL CA certificate is valid for one year. You must update the validity period of the SSL CA certificate in your application or client within one year. Otherwise, your application or client that uses

encrypted network connections cannot connect to the ApsaraDB RDS instance.

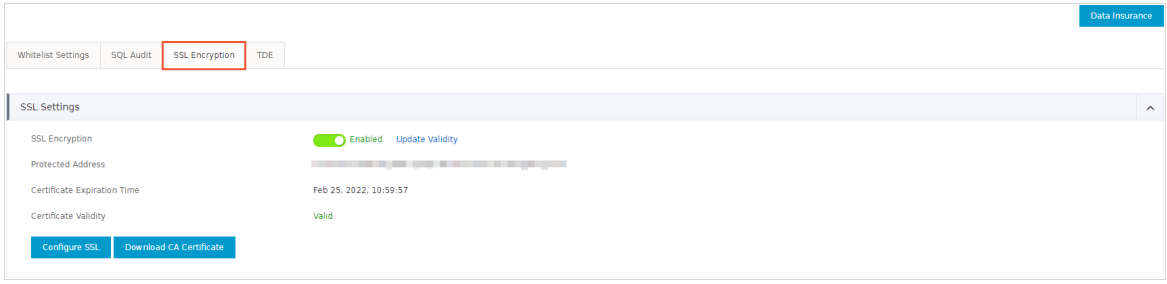
- SSL encryption may cause a significant increase in CPU utilization. We recommend that you enable SSL encryption only when you want to encrypt connections from the Internet. In most cases, connections that use an internal endpoint do not require SSL encryption.
- Read/write splitting endpoints do not support SSL encryption.
- If you disable SSL encryption, the ApsaraDB RDS instance restarts. Proceed with caution.

Enable SSL encryption

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **SSL Encryption** tab.



5. In the **SSL Settings** section, turn on **SSL Encryption**.
6. In the **Configure SSL** dialog box, select the endpoint for which you want to enable SSL encryption and click **OK**.
7. Click **Download CA Certificate** to download the SSL CA certificate files in a compressed package.



The downloaded package contains the following files:

- P7B file: contains the server CA certificate that can be imported into a Windows operating system.
- PEM file: contains the server CA certificate that can be imported into an operating system other than Windows or an application that is not Windows-based.
- JKS file: contains the server CA certificate that is stored in a Java-supported truststore. You can use the file to import the CA certificate chain into a Java-based application. The default password is apsaradb.

Note When the JKS file is used in Java, you must modify the default JDK security configuration in JDK 7 and JDK 8. Open the `/jre/lib/security/java.security` file on the host where your application resides and modify the following configurations:

```
jdk.tls.disabledAlgorithms=SSLv3, RC4, DH keySize < 224
jdk.certpath.disabledAlgorithms=MD2, RSA keySize < 1024
```

If you do not modify the JDK security configuration, the following error is reported. Similar errors are also caused by the Java security configuration.

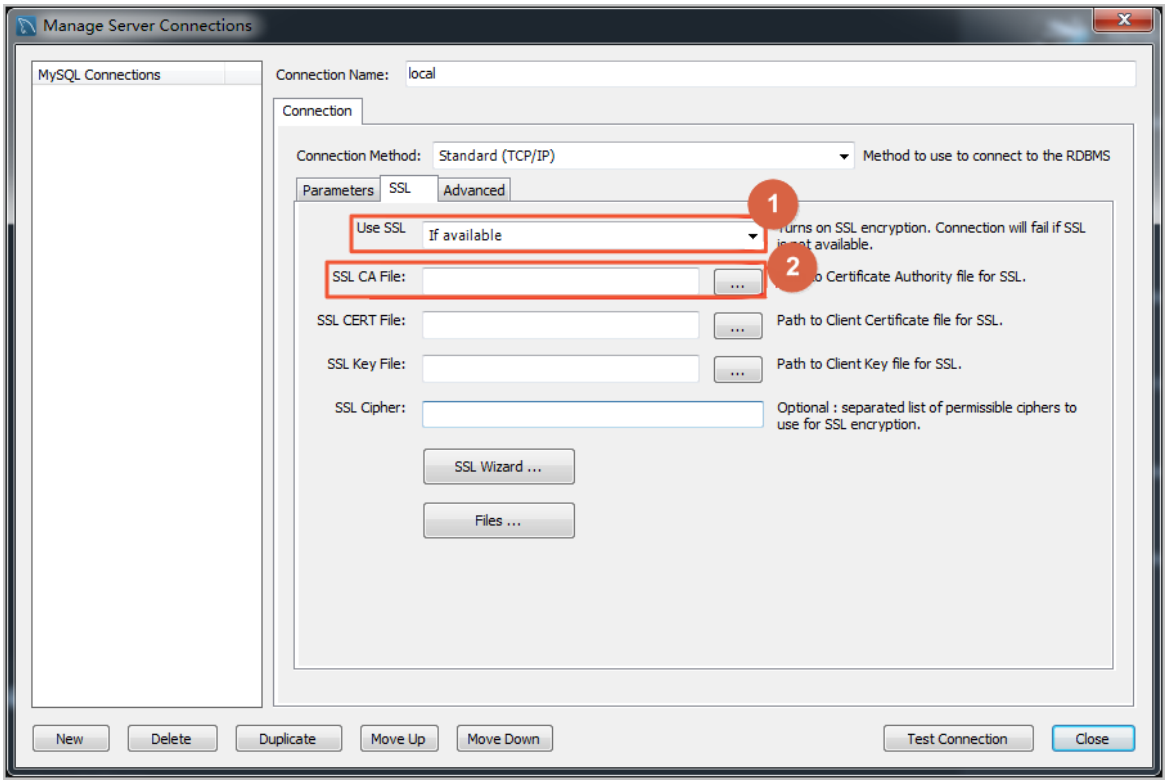
```
javax.net.ssl.SSLHandshakeException: DHPublicKey does not comply to algorithm con
straints
```

Configure an SSL CA certificate

After you enable SSL encryption, configure the SSL CA certificate on your application or client before they can connect to the ApsaraDB RDS instance. This section describes how to configure an SSL CA certificate. MySQL Workbench and Navicat are used in the example. If you are using other applications or clients, see the related instructions.

Configure a certificate on MySQL Workbench

1. Start MySQL Workbench.
2. Choose **Database > Manage Connections**.
3. In the **Connection** section, click the **SSL** tab and configure the following parameters.



①: Enable **Use SSL**.

②: Import the SSL CA certificate file.

Configure a certificate on Navicat

1. Start Navicat.
2. Right-click the database and select **Edit Connection**.
3. Click the **SSL** tab. Select the path of the PEM-formatted CA certificate, as shown in the following figure.
4. Click **OK**.

> **Note** If the `connection is being used` error is reported, the previous session is still connected. Restart Navicat.

5. Double-click the database to test whether the database is connected.

Update the validity period of an SSL CA certificate

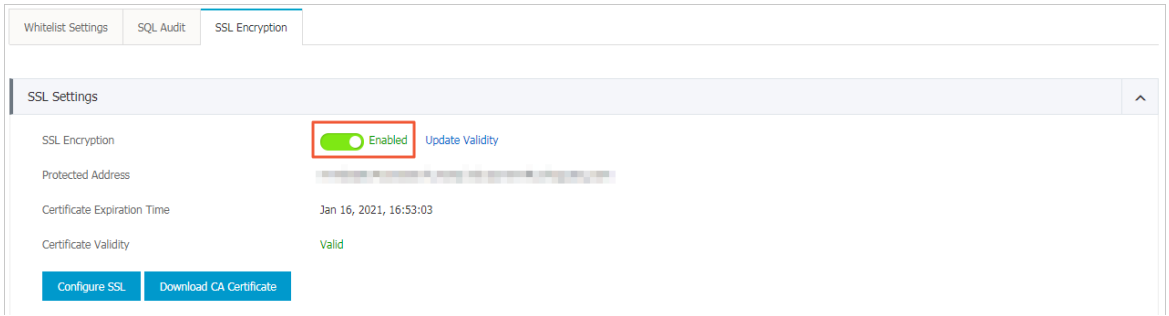
> **Note** **Update Validity** causes the ApsaraDB RDS instance to restart. Proceed with caution.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **SSL Encryption** tab.
5. Click **Update Validity**.

Disable SSL encryption

> **Note**
> - If you disable SSL encryption, the ApsaraDB RDS instance restarts. To reduce the impact on your business, the system triggers a primary/secondary switchover. We recommend that you disable SSL encryption during off-peak hours.
> - After you disable SSL encryption, access performance increases, but security decreases. We recommend that you disable SSL encryption only in secure environments.

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **SSL Encryption** tab.
5. In the **SSL Settings** section, turn off **SSL Encryption**. In the message that appears, click **OK**.



10.3. Configure TDE

This topic describes how to configure Transparent Data Encryption (TDE) for an ApsaraDB RDS instance. TDE encrypts and decrypts data files in real time. It encrypts data files when they are written to disks, and decrypts data files when they are loaded to the memory from disks. TDE does not increase the size of data files. You can use TDE without the need to make changes to applications.

Prerequisites

- Your ApsaraDB RDS instance runs the RDS High-availability Edition with local SSDs.
- Key Management Service (KMS) is activated. If KMS is not activated, you can activate it when you enable TDE.

Context

The key used for TDE is created and managed by KMS. ApsaraDB RDS does not provide the key or certificates that are required for encryption. For specific zones, you can use the keys that are automatically generated by Apsara Stack, or you can use your own key materials to generate data keys and authorize your ApsaraDB RDS instance to use these keys.

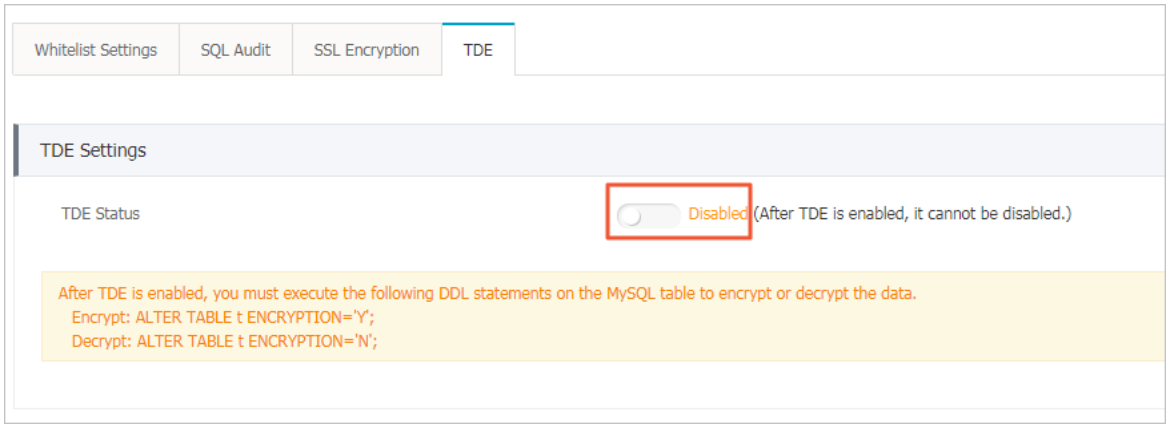
Precautions

- When TDE is being enabled, your ApsaraDB RDS instance is restarted and all services are disconnected. Make appropriate service arrangements before you enable TDE. Proceed with caution.
- You cannot disable TDE after it is enabled.
- You cannot change the key used for encryption after TDE is enabled.
- If you want to restore the data to your computer after TDE is enabled, you must decrypt data on your ApsaraDB RDS instance. For more information, see the "Decrypt a table" section of this topic.
- After TDE is enabled, CPU utilization significantly increases.
- If you use an existing custom key for encryption, take note of the following items:
  - If you disable the key, configure a plan to delete the key, or delete the key material, the key becomes unavailable.
  - If you revoke the key that is authorized for an ApsaraDB RDS instance, the instance becomes unavailable after it is restarted.
  - You must use an Apsara Stack tenant account or an account that has the AliyunSTSAssumeRoleAccess permission.

> **Note** For more information, see topics about key management in *Key Management Service User Guide*.

Use a key that is automatically generated by Apsara Stack

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **TDE** tab.
5. In the **TDE Settings** section, turn on **TDE Status**.

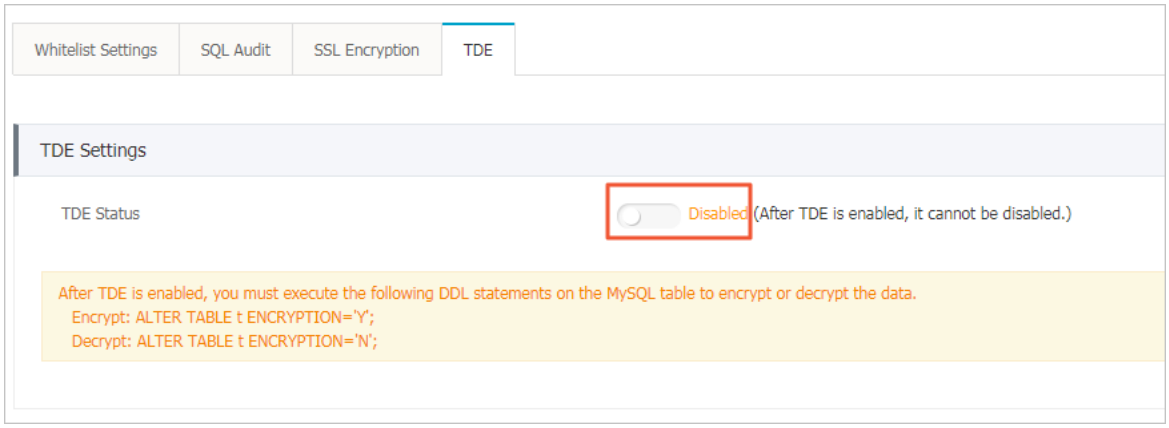


6. In the dialog box that appears, select **Use an Automatically Generated Key** and click **OK**.

> **Note** If the instance runs MySQL 5.7 on RDS High-availability Edition, you can select one of the following encryption methods:
>
> - SM4 encryption
> - AES_256_CBC encryption

Use an existing custom key

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **TDE** tab.
5. In the **TDE Settings** section, turn on **TDE Status**.



6. In the dialog box that appears, select **Use an Existing Custom Key** and click **OK**.

> **Note** If you do not have a custom key, click **create a key** to go to the KMS console and import the key materials. For more information, see Create a key in *Key Management Service User Guide*.

Encrypt a table

Log on to the database and execute one of the following statements to encrypt a table:

- MySQL 5.6

```
alter table <tablename> engine=innodb,block_format=encrypted;
```

- MySQL 5.7 or MySQL 8.0

```
alter table <tablename> encryption='Y';
```

Decrypt a table

Execute one of the following statements to decrypt a table that is encrypted by using TDE:

- MySQL 5.6

```
alter table <tablename> engine=innodb,block_format=default;
```

- MySQL 5.7 or MySQL 8.0

```
alter table <tablename> encryption='N';
```

FAQ

- Q: After I enable TDE, can I still use common database tools such as Navicat?

  A: Yes, after you enable TDE, you can still use common database tools such as Navicat.

- Q: After I enable TDE, why is my data still in plaintext?

  A: After you enable TDE, your data is stored in ciphertext. However, when the data is queried, it is decrypted and loaded into memory as plaintext. A: TDE encrypts backup files to prevent data leaks. Before you restore the data of your ApsaraDB RDS instance from an encrypted backup file to your computer, you must decrypt the file. For more information, see the "Decrypt a table" section of this topic.

10.4. Configure SQL audit

You can use the SQL audit feature to audit SQL executions and view their details. The SQL audit feature does not affect instance performance.

Context

> **Note** You cannot view the logs that are generated before you enable SQL audit.

You can view the incremental data of your ApsaraDB RDS for MySQL instance in SQL audit logs or binlogs. However, these two methods differ in the following aspects:

- SQL audit logs are similar to audit logs in MySQL and record all DML and DDL operations by using network protocol analysis. SQL audit does not parse the actual parameter values. Therefore, a small amount of information may be lost if a large number of SQL statements are executed to query data. The incremental data obtained by using this method may be inaccurate.
- Binlogs record all add, delete, and modify operations and the incremental data used for data restoration. Binlogs are temporarily stored in your ApsaraDB RDS instance after they are generated. The system transfers full binlog files to Object Storage Service (OSS) on a regular basis. OSS then stores the files for seven days. However, a binlog file cannot be transferred if data is being written to it. Such binlog files cannot be uploaded to OSS after you click **Upload Binlogs** on the **Backup and**

Restoration page. Binlogs are not generated in real time, but you can obtain accurate incremental data from them.

Precautions

- By default, SQL audit is enabled.
- SQL audit logs are retained for 7 days.
- Log files exported from SQL audit are retained for two days. The system clears files that are retained for longer than two days.

Disable SQL audit

> **Note** If SQL audit is disabled, all SQL audit logs are deleted. We recommend that you export and store audit logs to your computer before you disable SQL audit.

You can disable SQL audit to avoid charges when you do not need it. To disable SQL audit, perform the following operations:

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **SQL Audit** tab.
5. Click **Export File** to export and store the SQL audit content to your computer.
6. After the file is exported, click **Disable SQL Audit**.
7. In the message that appears, click **Confirm**.

Enable SQL audit

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Data Security**.
4. Click the **SQL Audit** tab.
5. Click **Enable SQL Audit**.
6. In the message that appears, click **Confirm**.

   After SQL audit is enabled, you can query SQL information based on conditions such as the time range, database, user, and keyword.

11.Service availability

11.1. Switch workloads over between primary and secondary ApsaraDB RDS instances

This topic describes how to switch workloads over between a primary ApsaraDB RDS instance and its secondary instance. ApsaraDB RDS supports both manual switchover and automatic switchover. After a switchover is complete, the primary ApsaraDB RDS instance becomes the secondary instance

Description

ApsaraDB RDS supports both manual switchover and automatic switchover.

- Automatic switchover: By default, the automatic switchover feature is enabled. If the primary ApsaraDB RDS instance becomes faulty, ApsaraDB RDS automatically switches workloads over to the secondary instance.
- Manual switchover: You can manually switch workloads over between the primary and secondary ApsaraDB RDS instances even when the automatic switchover feature is enabled.

Precautions

- Data is synchronized between the primary and secondary ApsaraDB RDS instances in real time. The secondary instance serves only as a read-only instance or standby. We recommend that you do not modify the data of the secondary instance. If you modify the data of the secondary instance, the data of the primary instance may be accidentally deleted or overwritten, which may irreparably impact your workloads.
- If the secondary ApsaraDB RDS instance serves as a read-only instance, the secondary instance serves as the new primary instance after a primary/secondary switchover. We recommend that you plan the read and write workloads for the new primary instance before the switchover.
- During a switchover, your service may be interrupted. Make sure that your application is configured to automatically reconnect to the instance.
- If the primary ApsaraDB RDS instance is attached with read-only instances, the read-only instances need to re-establish the connections that are used for data replication and synchronize incremental data after a switchover. As a result, data synchronization to the read-only instances has a latency of a few minutes.

Perform a manual primary/secondary switchover

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Service Availability**.
4. Click **Switch Primary/Secondary Instance** on the right side of the page.

> **Note** You may encounter a service interruption during a switchover. Make sure that your application is configured to automatically reconnect to the instance.

5. In the dialog box that appears, click **OK**.

> **Note** In the dialog box, you can also select **Switch Within Maintenance Window** and click **OK**. Then, the system performs the primary/secondary switchover within the maintenance window. For more information about how to set the maintenance window, see Set a maintenance window. You can also click **Change** on the right to change the maintenance window.

FAQ

Q: Can I connect to secondary instances?

A: No, you cannot connect to secondary instances in the same manner as you connect to the primary instance. Secondary instances serve only as a read-only instance or standby. We recommend that you do not modify the data of secondary instances. If you modify the data of secondary instances, the data of the primary instance may be accidentally deleted or overwritten, which may irreparably impact your workloads.

11.2. Change the data replication mode

You can set the data replication mode between primary and secondary ApsaraDB RDS instances to improve database availability.

Prerequisites

Your ApsaraDB RDS instance runs the RDS High-availability Edition.

Data replication modes

- Semi-synchronous

  After an update that is initialized by your application is complete on the primary instance, the log is synchronized to all the secondary instances. After the secondary instances receive the log, the update transaction is considered committed. Your database system does not need to wait for the log to be replayed.

  If the secondary instances are unavailable or a network exception occurs between the primary and secondary instances, semi-synchronous replication degrades to the asynchronous mode.

- Asynchronous

  When your application initiates a request to add, delete, or modify data, the primary instance responds to your application immediately after it completes the operation. At the same time, the primary instance starts to asynchronously replicate data to its secondary instances. During asynchronous data replication, the unavailability of secondary instances does not affect the operations on the primary instance. Data remains consistent even if the primary instance is unavailable.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.

3. In the left-side navigation pane, click **Service Availability**.
4. Click **Change Data Replication Mode** on the right side of the page.
5. In the dialog box that appears, select a data replication mode and click **OK**.

FAQ

Q: Which data replication mode is recommended?

A: You can select a data replication mode based on your business requirements. If you require quick responses, we recommend that you select the asynchronous mode. In other scenarios, you can select the semi-synchronous mode.

11.3. Configure forced failover

If both the primary or secondary instance and the logger instance fail for an ApsaraDB RDS Enterprise Edition instance, the forced failover feature allows you to activate the available primary or secondary instance to become the standalone primary instance. This operation can recover workloads in a short period of time.

Prerequisites

An ApsaraDB RDS Enterprise Edition instance is created. For more information about how to create an instance, see Create an instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Service Availability**.
4. In the **Availability Information** section of the page that appears, click **Forced Failover**.
5. In the Forced Failover dialog box, select a failover method based on the states of primary, secondary, and logger instances and click **OK**. Note: Forced failover cannot ensure a recovery point objective (RPO) of zero to retain all data. Proceed with caution.

> **Note** Forced failover activates the available primary or secondary instance to become the standalone primary instance and disassociates it from the other two instances. This operation can immediately recover workloads. Later, the deployment mode of three nodes automatically resumes.

12.Database backup and restoration

12.1. Configure automatic backup

Automatic backup of ApsaraDB RDS supports full physical backups. ApsaraDB RDS automatically backs up data based on pre-configured policies. This topic describes how to configure a policy for automatic backup.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Backup and Restoration**.
4. Click the **Backup Settings** tab.
5. Click **Edit**. In the dialog box that appears, configure the parameters described in the following table.

> **Note** To ensure data security, the system compares the new backup cycle and time with the original settings and selects the most recent point in time to back up the data. Therefore, the next backup may still be performed based on the original backup cycle and time. For example, assume that the scheduled backup cycle and time is set to 19:00-20:00 every Wednesday. If you modify the backup cycle and time to 19:00-20:00 every Thursday before the scheduled backup occurs, the system still backs up data at the original scheduled time of 19:00-20:00 of Wednesday.

Parameter	Description
Data Retention Period	The number of days for which data backup files are retained. Valid values: 7 to 730. Default value: 7.
Backup Cycle	The backup cycle. You can select one or more days within a week.
Backup Time	A period of time within a day. Unit: hours. We recommend that you back up data during off-peak hours.
Log Backup	Specifies whether to enable log backup. **Notice** If you disable log backup, all the log backup files are deleted and you cannot restore data to a specific point in time.
Log Retention Period	The number of days for which log backup files are retained. Valid values: 7 to 730. Default value: 7.

6. Click **OK**.

12.2. Manually back up an instance

Manual backup of ApsaraDB RDS supports both full physical and logical backups. This topic describes how to manually back up an ApsaraDB RDS instance.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. Click **Back Up Instance** in the upper-right corner.
4. In the dialog box that appears, set the backup mode and backup policy and then click **OK**.

> **Note** The following backup modes are available:
>
> - Physical backup: directly backs up all files in all databases.
> - Logical backup: extracts data from the databases by using SQL statements and backs up the data in the text format. If you select logical backup, you must select one of the following backup policies:
>   - Instance Backup: backs up the entire instance.
>   - Single-Database Backup: backs up one of the databases on the instance.

12.3. Download data and log backup files

This topic describes how to download unencrypted data and log backup files in the ApsaraDB RDS console to archive the files and restore data to an on-premises database.

Limits

Database engine	Download of data backup files	Download of log backup files
MySQL 5.6 on RDS High-availability Edition	Supported	Supported
MySQL 5.7 on RDS High-availability Edition or Enterprise Edition	Supported	Supported

Database engine	Download of data backup files	Download of log backup files
MySQL 8.0 on RDS High-availability Edition	Supported	Supported

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Backup and Restoration**.
4. Click the Data Backup or Log Backup tab.
   - To download data backup files, click the **Data Backup** tab.
   - To download log backup files, click the **Log Backup** tab.
5. Select a time range to which you want to restore the instance.
6. Find the data or log backup file that you want to download, and click **Download** in the **Actions** column.

   > **Note**
   > - If the Download button is unavailable, see the "Limits" section of this topic.
   > - If you want to use a data backup file to restore data, select the backup file that is the closest to the time for restoration.
   > - If you want to use a log backup file to restore data to an on-premises database, take note of the following items:
   >   - The instance No. of the log backup file must be the same as that of the data backup file.
   >   - The start time of the log backup file must be later than the end time of the selected data backup file and earlier than the point in time to which you want to restore the data of your instance.

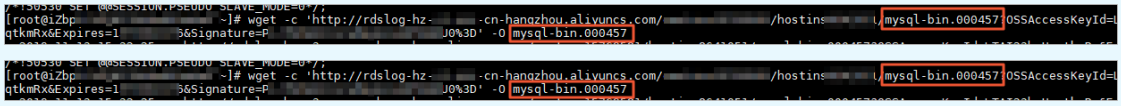
7. In the message that appears, click **Download**.

Download method	Description
Download	Use a browser to download the backup file.
Copy Internal URL	Copy the internal URL to download the file. If your Elastic Compute Service (ECS) and ApsaraDB RDS instances reside within the same region, you can log on to the ECS instance and use the internal URL to download the file. This method is fast and secure.
Copy Public URL	Copy the public URL to download the file. If you want to use other tools to download the file, use the public URL.

Note If you use a Linux operating system, you can run the following command to download the file:

```
wget -c '<The URL used to download the backup file>' -O <The name of the backup fil
e>
```

- The -c option enables resumable download.
- The -O option saves the downloaded file by using the specified name. We recommend that you use the file name contained in the download URL.
- If the URL contains more than one parameter, enclose the download URL in a pair of single quotation marks (').



12.4. Upload binlogs

Context

This topic describes how to upload binlog files to Object Storage Service (OSS).

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Backup and Restoration** to go to the **Backup and Restoration** page.
4. In the upper-right corner of the page, click **Upload Binlogs**.
5. In the message that appears, click **Confirm**.

12.5. Restore data to a new instance (formerly known as cloning an instance)

A cloned instance is a new instance that has the same content as the primary instance, including data and settings. This feature allows you to restore data of the primary instance or create multiple instances that are the same as the primary instance.

Prerequisites

Before you clone an instance, make sure that the following requirements are met:

- The primary instance is in the running state.
- The primary instance does not have ongoing migration tasks.
- Data backup and log backup are enabled.
- The primary instance has at least one completed backup set before you clone the instance by

backup set.

Context

You can specify a backup set or a point in time within the backup retention period to clone an instance.

> **Note**
>
> - A cloned instance copies only the content of the primary instance, but not the content of read-only instances. The copied data includes database information, account information, and instance settings such as whitelist settings, backup settings, parameter settings, and alert threshold settings.
> - The database engine of a cloned instance must be the same as that of the primary instance. Other settings can be different, such as the instance edition, zone, network type, instance type, and storage capacity. If you want to restore the data of a primary instance, we recommend that you select a higher instance type and more storage capacity than those of the primary instance. This can speed up the data restoration process.
> - The account type of a cloned instance must be the same as that of the primary instance. The account password of the cloned instance can be changed.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Backup and Restoration**.
4. In the backup list, find a backup and click **Restore** in the Actions column.
5. In the dialog box that appears, select **Restore Database** and click **OK**.
6. On the **Restore Instance** page, configure the following parameters.

Section	Parameter	Description
Region	**Region**	The region in which the cloned instance resides.
Restore Database	**Restore Mode**	The data restore mode of the primary instance. Valid values: ○ **By Time** ○ **By Backup Set**
	Restore Time	The point in time to which you want to restore the primary instance. **Note** When **Restore Mode** is set to **By Time**, you must specify this parameter.

Section	Parameter	Description
	Backup Set	The backup set for restoration. **Note** When **Restore Mode** is set to **By Backup Set**, you must specify this parameter.
Specifications	**Instance Name**	The name of the cloned instance.
	Database Engine	The database engine of the cloned instance, which cannot be modified.
	Engine Version	The engine version of the cloned instance, which cannot be modified.
	Edition	The RDS edition of the cloned instance. The actual values are displayed in the console.
	Storage Type	The storage type of the cloned instance. The actual values are displayed in the console.
	Instance Type	The instance type of the cloned instance. **Note** We recommend that you select a higher instance type and more storage capacity than those of the primary instance. This can speed up the data restoration process.
	Storage Capacity	The storage capacity of the cloned instance, which includes the space to store data, system files, binlog files, and transaction files. The available storage capacity is displayed in the console. **Note** AsparaDB RDS instances with local SSDs in the dedicated instance family occupy exclusive resources. The storage capacities are determined based on instance types.

Section	Parameter	Description
Network Type	**Network Type**	The network type of the cloned instance. ApsaraDB RDS instances support the following network types: ○ **Classic Network**: Cloud services in the classic network are not isolated. Unauthorized access to a cloud service is blocked only by the security group or whitelist policy of the service. ○ **VPC**: A virtual private cloud (VPC) helps you build an isolated network environment on Apsara Stack. You can customize route tables, CIDR blocks, and gateways within a VPC. We recommend that you select VPC for improved security.
	VPC	The VPC in which the cloned instance resides. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.
	vSwitch	The vSwitch in the VPC. **Note** When **Network Type** is set to **VPC**, you must specify this parameter.

7. Click **Submit**.

13.CloudDBA

13.1. Introduction to CloudDBA

CloudDBA is a cloud service for database self-detection, self-repair, self-optimization, self-maintenance, and self-security based on machine learning and expertise. CloudDBA helps you ensure stable, secure, and efficient databases without worrying about the management complexity and service failures caused by manual operations.

Features

In ApsaraDB RDS for MySQL, CloudDBA provides the following features:

- Diagnostics

  You can diagnose your instance and view the visualized diagnostic results.

- Autonomy center

  You can configure automatic detection for exceptions on core metrics. When an exception is detected, the system performs diagnostics on sessions, SQL statements, and the database capacity, provides optimization suggestions, and then performs automatic optimization if the related permissions have been granted.

- Instance sessions

  You can view sessions, check session statistics, analyze SQL statements, and optimize the execution of SQL statements.

- Real-time monitoring

  You can view the real-time information of your instance, such as the queries per second (QPS), transactions per second (TPS), number of connections, and network traffic.

- Storage analysis

  You can view the storage overview, trends, exceptions, tablespaces, and data spaces.

- Deadlock analysis

  You can view and analyze the last deadlock in a database.

- Dashboard

  You can view and compare performance trends, customize monitoring dashboards, check exceptions, and view instance topologies.

- Slow query logs

  You can view the trends and statistics of slow queries.

- Diagnostic reports

  You can generate diagnostic reports or view automatically generated reports about instance health, alerts, and slow queries.

13.2. Diagnostics

In ApsaraDB RDS for MySQL, CloudDBA provides the diagnostics feature. This feature diagnoses your ApsaraDB RDS for MySQL instance and visualizes the results.

Navigate to the Diagnostics tab

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Diagnostics** tab.

> **Note** For more information, see Diagnostics in *Database Autonomy Service User Guide*.

13.3. Autonomy center

In ApsaraDB RDS for MySQL, CloudDBA provides the autonomy center feature. When an exception on core metrics is detected by CloudDBA, the system performs diagnostics on sessions, SQL statements, and the database capacity to identify the causes. CloudDBA also provides optimization and mitigation suggestions. If the related permissions have been granted, CloudDBA automatically performs the optimization and mitigation operations.

Navigate to the Autonomy Center tab

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Autonomy Center** tab.

> **Note** For more information, see Diagnostics in *Database Autonomy Service User Guide*.

13.4. Session management

In ApsaraDB RDS for MySQL, CloudDBA provides the session management feature. This feature allows you to view and manage the sessions of an instance.

Navigate to the Session Management page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Session Management** tab.

> **Note** For more information, see Instance sessions in *Database Autonomy Service User Guide*.

13.5. Real-time monitoring

In ApsaraDB RDS for MySQL, CloudDBA provides the real-time monitoring feature. This feature allows you to view the real-time performance of your ApsaraDB RDS for MySQL instance.

Navigate to the Real-time Monitoring tab

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Real-time Monitoring** tab.

> **Note** For more information, see Real-time monitoring in *Database Autonomy Service User Guide*.

13.6. Storage analysis

In ApsaraDB RDS for MySQL, CloudDBA provides the storage analysis feature. This feature allows you to check and solve storage exceptions in a timely manner to ensure database stability.

Context

You can use the storage analysis feature of CloudDBA to view the disk space usage of your ApsaraDB RDS for MySQL instance and the number of remaining days when disk space is available. It also provides information about the space usage, fragmentation, and exception diagnostic results of a table.

Navigate to the Storage Analysis tab

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Storage Analysis** tab.

> **Note** For more information, see Storage analysis in *Database Autonomy Service User Guide*.

13.7. Deadlock analysis

In ApsaraDB RDS for MySQL, CloudDBA provides the deadlock analysis feature. This feature allows you to view and analyze the last deadlock in a database.

Navigate to the Deadlock Analysis page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostics**.
4. Click the **Deadlock Analysis** tab.

> **Note** For more information, see Deadlock analysis in *Database Autonomy Service User Guide*.

13.8. Dashboard

In ApsaraDB RDS for MySQL, CloudDBA provides the dashboard feature. This feature allows you to view performance trends in specific ranges, compare performance trends, and customize charts to view performance trends.

Navigate to the Dashboard page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Dashboard**.

> **Note** For more information, see Dashboard in *Database Autonomy Service User Guide*.

13.9. Slow query logs

In ApsaraDB RDS for MySQL, CloudDBA provides the slow query logs feature. This feature allows you to view the trends and execution details of slow queries and obtain optimization suggestions for your ApsaraDB RDS for MySQL instance.

Navigate to the Slow Query Logs page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Slow Query Logs**.

> **Note** For more information, see Slow query logs in *Database Autonomy Service User Guide*.

13.10. Diagnostic reports

In ApsaraDB RDS for MySQL, CloudDBA provides the diagnostic reports feature. This feature allows you to create and view diagnostic reports.

Navigate to the Diagnostic Reports page

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, choose **CloudDBA > Diagnostic Reports**.

> **Note** For more information, see Diagnostic reports in *Database Autonomy Service User Guide*.

14.Manage logs

All ApsaraDB RDS instances support log management. You can query the details of error and slow query logs of an ApsaraDB RDS instance by using the ApsaraDB RDS console. The logs help you perform troubleshooting.

Procedure

1. Log on to the ApsaraDB for RDS console.
2. On the **Instances** page, find the instance that you want to manage.
3. In the left-side navigation pane, click **Logs**.
4. On the **Logs** page, click the **Error Logs**, **Slow Query Logs**, **Slow Query Log Summary**, or **Primary/Secondary Switching Logs** tab, select a time range, and then click **Search**.

Log type	Description
Error Logs	Records database running errors that occurred within the latest month.
Slow Query Logs	Records SQL statements within the last 15 days that took longer than one second to execute. Duplicated SQL statements are removed. **Note** Slow query logs in the ApsaraDB RDS console are updated once every minute. However, you can query real-time slow query logs from the mysql.slow_log table.
Slow Query Log Summary	Records and analyzes SQL statements within the last month that took longer than one second to execute. Analysis reports of slow query logs are provided.
Primary/Secondary Switching Logs	Records the primary/secondary instance switching logs. This feature is available for ApsaraDB RDS instances that run MySQL on RDS High-availability Edition.

15.Use mysqldump to migrate MySQL data

This topic describes how to use mysqldump to migrate data from an on-premises database to an ApsaraDB RDS for MySQL instance.

Prerequisites

An ECS instance is created.

Context

mysqldump is easy to use but requires extensive downtime. This tool is suitable for scenarios where the amount of data is small or extensive downtime is allowed.

ApsaraDB RDS for MySQL is fully compatible with the native database service. The procedure of migrating data from the original database to an ApsaraDB RDS for MySQL instance is similar to that of migrating data from one MySQL server to another.

Before you migrate data, you must create an account that is used to migrate data from the on-premises MySQL database. You must grant the read and write permissions on the on-premises MySQL databases to the account.

Procedure

1. Run the following command to create a migration account for the on-premises database:

   ```
   CREATE USER 'username'@'host' IDENTIFIED BY 'password';
   ```

   Parameter description:

   - username: the name of the account to be created.
   - host: the host from which the account is authorized to log on to the on-premises MySQL database. If you want to allow access from a local host, set this parameter to localhost. If you want to allow access from all hosts, set this parameter to a percent sign (%).
   - password: the password of the account.

   For example, you can run the following command to create an account with the username William and the password Changme123. The account is authorized to log on to the on-premises MySQL database from all hosts.

   ```
   CREATE USER 'William'@'%' IDENTIFIED BY 'Changme123';
   ```

2. Run the following command to grant permissions to the migration account in the on-premises database:

   ```
   GRANT SELECT ON databasename.tablename TO 'username'@'host' WITH GRANT OPTION; GRANT REPLICATION SLAVE ON databasename.tablename TO 'username'@'host' WITH GRANT OPTION;  GRANT REPLICATION SLAVE ON databasename.tablename TO 'username'@'host' WITH GRANT OPTION;
   ```

   Parameter description:

   - privileges: the operation permissions granted to the account, such as SELECT, INSERT, and UPDATE. To authorize the account to perform all operations, set this parameter to ALL.
   - databasename: the name of the on-premises MySQL database. If you want to grant all database permissions to the account, set this parameter to an asterisk (*).

- tablename: the name of the table whose data you want to migrate. If you want to grant all table permissions to the account, set this parameter to an asterisk (*).
- username: the name of the account.
- host: the host from which the account is authorized to log on to the on-premises MySQL database. If you want to allow access from a local host, set this parameter to localhost. If you want to allow access from all hosts, set this parameter to a percent sign (%).
- WITH GRANT OPTION: authorizes the account to use the GRANT statement. This parameter is optional.

For example, you can execute the following statement to grant all permissions on tables and databases to the William account. The account is authorized to log on to the database from all hosts.

```
GRANT ALL ON *. * TO 'William'@'%';
```

3. Use the data export tool of mysqldump to export data from the database as a data file.

> **Notice** Do not update data during data export. In this step, only data is exported. Stored procedures, triggers, and functions are not exported.

```
mysqldump -h localIp -u userName -p --opt --default-character-set=utf8 --hex-blob dbName
--skip-triggers > /tmp/dbName.sql
```

Parameter description:

- localIp: the IP address of the host where the on-premises MySQL database resides.
- userName: the account that is used to migrate data from the on-premises MySQL database.
- dbName: the name of the on-premises MySQL database.
- /tmp/dbName.sql: the name of the exported data file.

4. Use mysqldump to export stored procedures, triggers, and functions.

> **Notice** Skip this step if no stored procedures, triggers, or functions are used in the database. When stored procedures, triggers, and functions are exported, you must remove the DEFINER to ensure compatibility with ApsaraDB RDS for MySQL.

```
mysqldump -h localIp -u userName -p --opt --default-character-set=utf8 --hex-blob dbName
-R | sed -e 's/DEFINER[ ]*=[ ]*[^*]*\*/\*/' > /tmp/triggerProcedure.sql
```

Parameter description:

- localIp: the IP address of the host where the on-premises MySQL database resides.
- userName: the account that is used to migrate data from the on-premises MySQL database.
- dbName: the name of the on-premises MySQL database.
- /tmp/triggerProcedure.sql: the name of the exported stored procedure file.

5. Upload the data file and stored procedure file to the ECS instance.

In this example, the files are uploaded to the following paths:

```
/tmp/dbName.sql
```

```
/tmp/triggerProcedure.sql
```

6. Log on to the ECS console and import both the data file and the stored procedure file to the

destination ApsaraDB RDS for MySQL instance.

> **Note** For information about how to log on to the ECS instance, see topics in the **Connect to an instance** section of ECS User Guide.

```
mysql -h intranet4example.mysql.rds.aliyuncs.com -u userName -p dbName < /tmp/dbName.sql
```

```
mysql -h intranet4example.mysql.rds.aliyuncs.com -u userName -p dbName < /tmp/triggerProcedure.sql
```

Parameter description:

- intranet4example.mysql.rds.aliyuncs.com: the endpoint of the ApsaraDB RDS for MySQL instance. An internal endpoint is used in this example.
- userName: the migration account of the ApsaraDB RDS for MySQL database.
- dbName: the name of the on-premises MySQL database from which you want to import data.
- /tmp/dbName.sql: the name of the data file that you want to import.
- /tmp/triggerProcedure.sql: the name of the stored procedure file that you want to import.