Alibaba Cloud

Apsara Stack Agility

Product Introduction

Product Version：**2009, Internal: V3.4.0**

Document Version：20210106

⊂−⊃ 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.

- 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.
- 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.
- 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.
- 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).

- 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.
- 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.
⑦ Note	A note indicates supplemental instructions, best practices, tips, and other content.	⑦ 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.Introduction to Apsara Stack Agility

1.1. What is Apsara Stack Agility?

This topic describes the definition, benefits, and platform features of Apsara Stack Agility.

Definition

A private cloud is a cloud computing system deployed on the premises of an enterprise by a cloud computing service provider. Cloud infrastructure, software, and hardware resources are deployed in the private cloud behind a firewall to allow internal users of the enterprise to share the resources of the data center. The private cloud can be managed by the enterprise itself or by a third party, and located within or outside the enterprise. Private clouds provide better privacy and exclusivity than public clouds.

Private clouds are divided into the following types based on enterprise scale or business requirements:

- Multi-tenant comprehensive private clouds for industries and large groups: an end-to-end cloud system created in a top-down manner. The system is designed to drive hyper-scale digital applications and meet IT requirements such as the continuous integration and development of DevOps applications and the operation support of production environments.

- Single-tenant basic private clouds for small and medium-sized enterprises and scenarios: a cloud system that can perform local computing tasks and host technical systems such as large-scale Software as a Service (SaaS) applications, industrial clouds, and large group clouds.

Apsara Stack

More and more enterprises migrate their IT infrastructure to the cloud, and they must consider construction requirements such as security compliance, reuse of existing data centers, and the benefits of a collocated data center. Some enterprises may prefer to use their own data centers but want to deliver a service experience that relies on large-scale cloud computing.

Alibaba Cloud Apsara Stack is an extension of Alibaba Cloud public cloud, which brings the public cloud technologies to private clouds. Apsara Stack delivers complete and customizable Alibaba Cloud software solutions and allows enterprises to experience the same hyper-scale cloud computing and big data products as those provided by Alibaba Cloud public cloud within their own data centers. Apsara Stack also provides enterprises with a consistent hybrid cloud experience where you can obtain IT resources and ensure business continuity.

Apsara Stack Agility

Small and medium-sized private clouds make up the majority of the private cloud market. Users tend to deploy private clouds on a small scale. Alibaba Cloud has launched an agile cloud application platform for enterprises to migrate their business to small and medium-sized private clouds. This platform is designed to provide an open, unified, and trusted cloud platform for enterprises, enhance their core competitiveness in the cloud market, and meet their diverse business requirements.

Apsara Stack Agility can be directly deployed and managed on an existing hardware base such as x86 architecture to provide secure and stable enterprise-level services. Apsara Stack Agility provides hybrid deployment of the base, Apsara, network, and storage components to reduce the required number of physical servers, improve resource utilization, and provide scalability of resources. Apsara Stack Agility

can reduce the number of management and control nodes and provide high availability and data

security at a lower cost.

Benefits

Apsara Stack Agility helps governments and enterprises digitally transform their businesses and

services based on a variety of products and services and the digitalization practices of Alibaba Group,

and in combination with the mature solutions and rich experience in various industries. Apsara Stack

Agility provides the following benefits:

- Elastic

  Combines all resources into a single supercomputer and flexibly scales out resources to minimize

  costs and maximize performance and stability.

- Agile

  Uses Internet and microservice integration to accelerate innovation.

- Digital

  Uses digitalization to allow data to flow vertically between businesses and forms a mid-end to

  handle large amounts of data.

- Smart

  Allows smart transformation of businesses globally and helps reinvent business models.

Platform features

Apsara Stack is an enterprise-level cloud platform. It has the following features:

- Software-defined platform: masks underlying hardware differences, enables resources to scale up

  or out, and does not affect the performance of upper-layer applications.

- Production-level reliability and security compliance: ensures the continuity and security of

  enterprise data.

- Centralized access management: isolates permissions of different roles to facilitate subsequent

  O&M management.

1.2. Why Apsara Stack Agility?

This topic describes the benefits of Apsara Stack Agility.

1.2.1. Unified distributed cloud operating system

Both Apsara Stack Agility and Alibaba Cloud public cloud are based on the Apsara distributed operating

system. The Apsara system provides underlying services such as storage, computing, and scheduling

for the upper-layer services.

The Apsara system is a hyper-scale universal operating system developed by Alibaba Cloud for use

both inside and outside China. The Apsara system connects millions of servers around the world to act

as a supercomputer and provides computing capabilities as online public services. The computing

capabilities provided by the Apsara system are powerful, universal, and accessible to everyone.

Apsara system kernel architecture
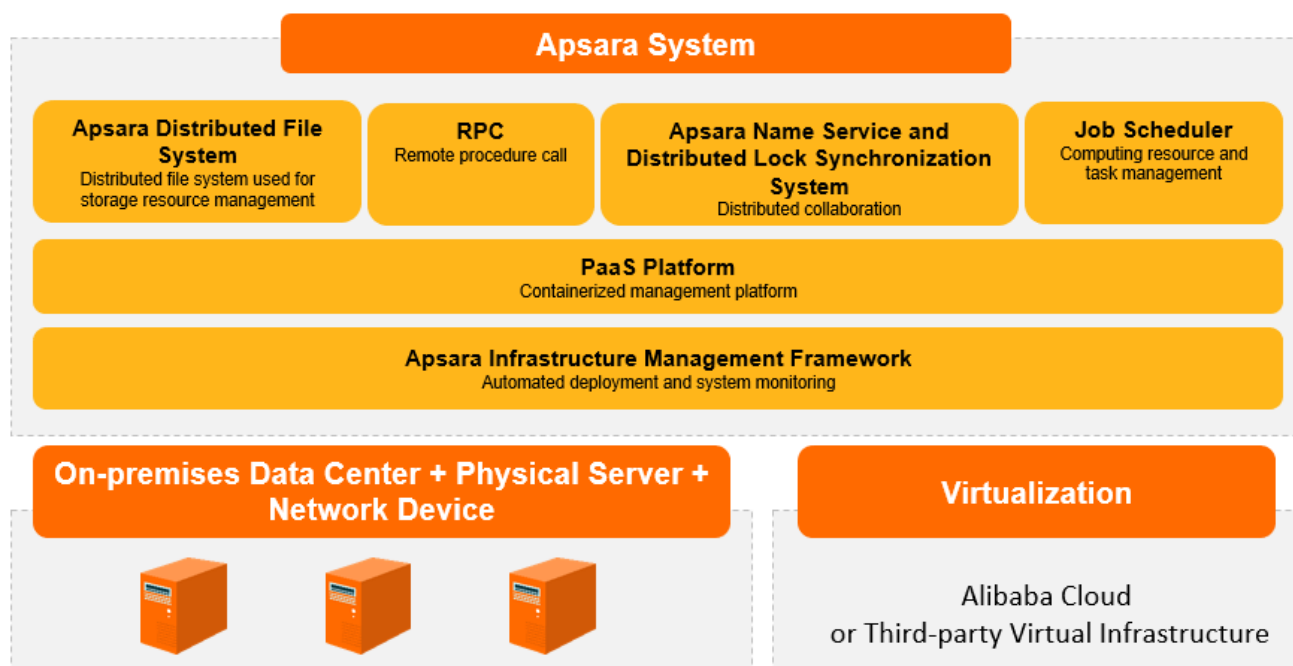
The Apsara system kernel consists of the following modules:

- Underlying services for distributed systems

  This module provides coordination, remote procedure call, security management, and resource

  management services needed in a distributed environment. These services provide support for

  upper-layer modules such as the distributed file system and task scheduling module.

- Distributed file system

  This module provides a reliable and scalable service to store large amounts of data. The

  distributed file system aggregates the storage capabilities of each node in a cluster and

  automatically protects against hardware and software faults to provide uninterrupted access to

  data. This module also supports incremental scaling and automatic data load balancing. An API

  similar to Portable Operating System Interface of UNIX (POSIX) is provided to access user space

  files. Additionally, the module supports random read/write and append write operations.

- Task scheduling

  This module schedules tasks in the cluster system and supports both online services that rely on a
  quick response speed and offline tasks that require high data processing throughput. The module
  can automatically detect faults and hot spots in the system. The module ensures stable and
  reliable service operations through such methods as error retry and concurrent backup for long-tail
  operations.

- PaaS platform

  This module automatically deploys and flexibly schedules application components on the
  underlying heterogeneous computing nodes. The module also provides unified O&M capabilities
  such as daily inspection, monitoring and alerting, and container service to ensure stable operation
  of all components on the PaaS platform.

- Cluster monitoring and deployment

  This module monitors the status of clusters as well as the running status and performance metrics
  of upper-layer application services, and generates alerts and records of exception events.
  Additionally, the module provides O&M personnel with deployment and configuration management
  of the entire Apsara system and its upper-layer applications. The module supports both the online
  elastic scaling of clusters and the online upgrade of application services.

1.2.2. Apsara Infrastructure Management Framework

This topic describes the modules of Apsara Infrastructure Management Framework and the functions of
each module.

Apsara Infrastructure Management Framework provides cloud services with underlying support

capabilities such as unified deployment, verification, authorization, and control. Apsara Infrastructure

Management Framework includes modules such as deployment framework, resource library,

metadatabase, authentication and authorization, API Gateway, and control service.

- The deployment framework provides unified deployment of access platforms and manages

  service dependencies.

- The resource library stores the executable files of all cloud services and their dependent

  components.

- The authentication and authorization module provides access control capabilities for cloud

  services.

- API Gateway provides a centralized API management platform for cloud services.

- The control service module monitors the basic health status of each cloud service and supports

  the Apsara Stack O&M system.

1.2.3. Centralized operation management and automated O&M capability

Apsara Stack Agility provides a centralized management portal. You can configure different

management permissions for different roles.

Apsara Stack Agility allows you to manage O&M tasks and customize your cloud resource console by

using OpenAPI. Apsara Stack Agility can be synchronized and integrated with the existing Information

Technology Infrastructure Library (ITIL) systems of enterprises.

Centralized O&M management

[MISSING IMAGE: Centralized O&M management, Centralized O&M management]

1.2.4. OpenAPI

Apsara Stack provides a wide range of SDKs and RESTful APIs on the OpenAPI platform.

OpenAPI provides flexible access to a variety of Apsara Stack Agility services. You can also use

OpenAPI to obtain the basic control information of Apsara Stack and integrate Apsara Stack with your

centralized control system.

1.3. Architecture

This topic describes the system, network, and security architectures, and base modules of Apsara

Stack Agility.

1.3.1. Types of private cloud architectures

Provide clouds provide cloud native and integrated cloud architectures.

- Cloud native architecture

  The cloud native architecture is derived from Internet-based open architecture. Based on a

  distributed system framework, the cloud native architecture was used originally for big data and

  web applications and later used to provide a range of basic services.

- Integrated cloud architecture

  The integrated cloud architecture focuses on the virtualization of computing services. Integrated

  cloud architecture is a breakthrough from traditional computing architecture developed by

  OpenStack and has become the most popular choice for private cloud architectures.

Apsara Stack Agility uses a cloud native architecture based on the proprietary operating systems,

distributed technologies, and products of Alibaba Cloud. Apsara Stack Agility uses a single architecture

for a variety of deployment environments to support all cloud products and services. This architecture

provides complete open capabilities and self-developed and controllable capabilities.

1.3.2. System architecture

Apsara Stack Agility provides a consistent operations and maintenance (O&M) management

experience and an enterprise-class cloud security architecture based on the OpenAPI model.

The system architecture of Apsara Stack Agility consists of the following layers.

System architecture of Apsara Stack Agility
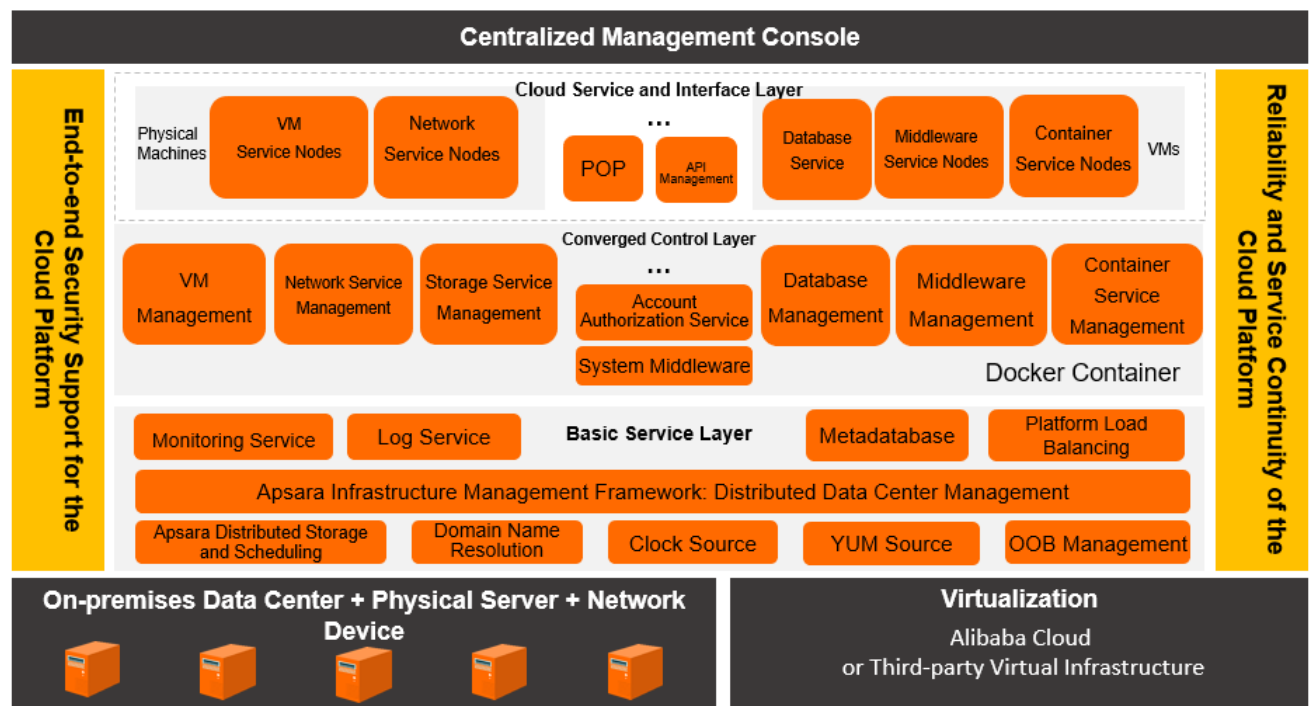
- Physical device layer: supports hardware devices such as data centers, x86 servers, networks,

  and Alibaba Cloud or third-party virtual infrastructure.

- Underlying service layer: provides services for upper-layer applications based on the underlying

  physical environment.

- Converged control layer: provides centralized scheduling for upper-layer application services based on a converged control architecture.

- Cloud service and interface layer: uses a converged management mechanism to provide centralized management and O&M for virtual and physical machines. The OpenAPI platform is used to provide centralized API management and support custom development.

- Centralized management layer: provides centralized O&M management.

Apsara Stack Agility also provides end-to-end security to ensure the reliability and service continuity of the cloud platform.

Logical architecture

Apsara Stack Agility virtualizes the computing and storage capabilities of hosts and network devices into virtual computing, distributed storage, and software defined networks (SDNs). Additionally, Apsara Stack Agility provides ApsaraDB and distributed middleware services as fundamental IT infrastructure support for your applications. Apsara Stack Agility can be integrated with your existing account, monitoring, and maintenance systems.

Logical architecture of Apsara Stack Agility (deployed on physical machines)
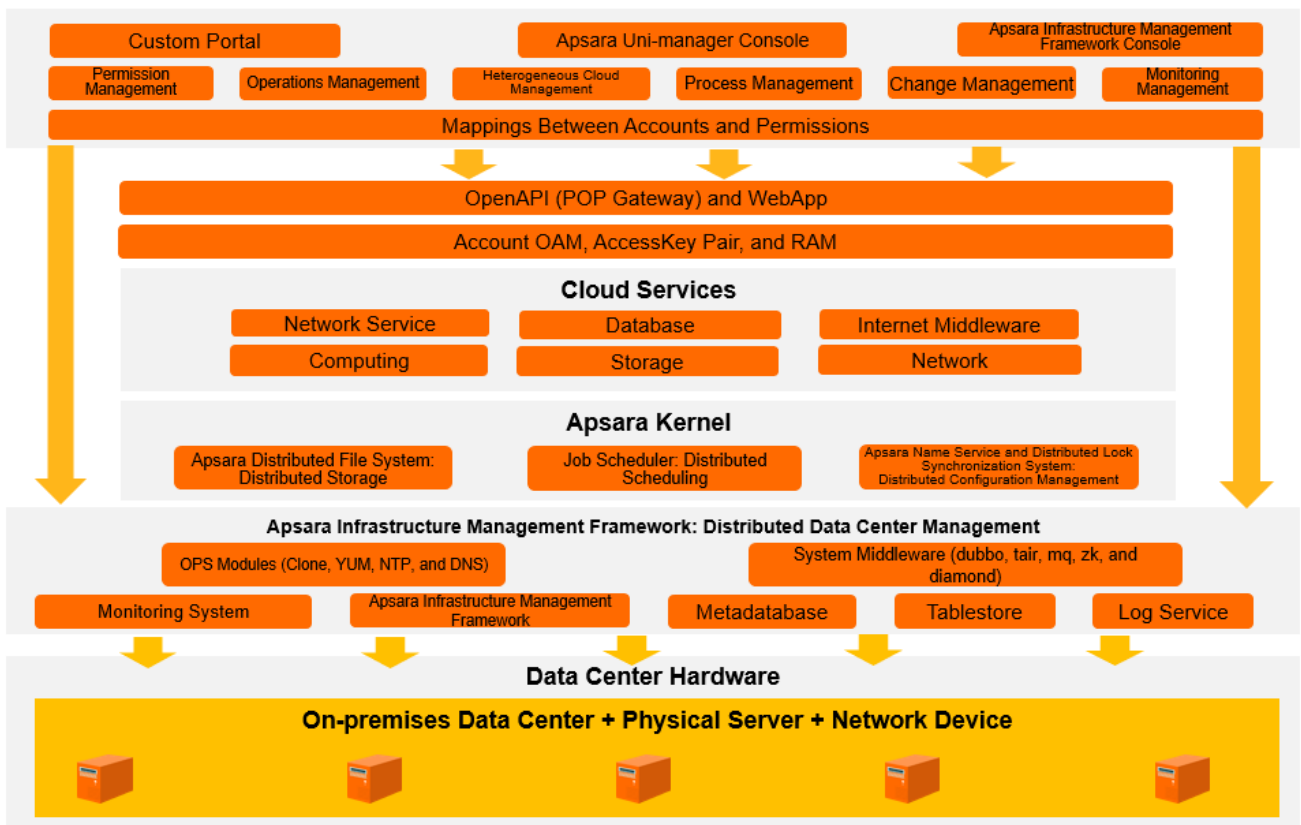
The logical architecture of Apsara Stack Agility has the following features:

- The hardware infrastructure of Apsara Stack Agility supports data centers, x86 servers, and

  networks.

- A variety of cloud services are provided based on the Apsara kernel (distributed engine).

- All cloud services are required to comply with a unified API framework, security system, and O&M

  and management system (including accounts, authorization, monitoring, and logs).

- A consistent user experience is guaranteed across all services.

Architectures deployed based on virtualization use a container solution to ensure a consistent

experience across cloud platforms. Underlying architectures deployed based on virtualization are the

same as those deployed on physical machines, except that bases of virtualization architectures are

deployed based on containers. No differences exist in user experiences and service features between

the two types of architectures.

Logical architecture of Apsara Stack Agility (deployed based on virtualization)
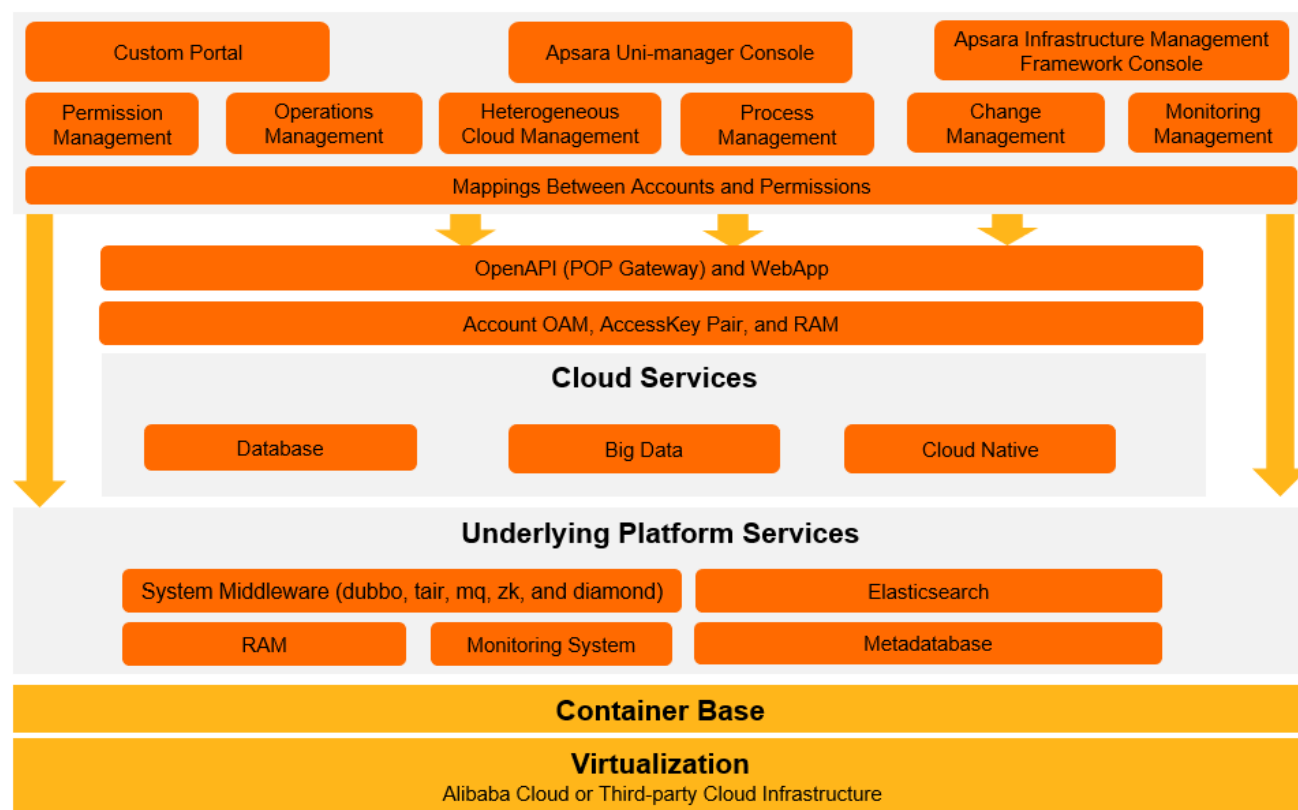
1.3.3. Network architecture

Apsara Stack Agility provides physical and virtual network architectures based on different scenarios.

1.3.3.1. Physical network architecture

Apsara Stack Agility is a lightweight version of Apsara Stack. The physical network architecture of

Apsara Stack Agility is optimized and streamlined to include only inter-connection switch (ISW) and

access switch (ASW) device roles. Apsara Stack Agility supports up to 96 servers, and uses MiniLVS in

place of Server Load Balancer (SLB) to support the Border Gateway Protocol (BGP).

The following table describes the roles and features of switches at different layers in the physical

network scenario.

Role definition

Role name	Description
ISW	The inter-connection switch. ISWs provide access to Internet service providers (ISPs) and are internally connected to ASWs.
ASW	The access switch. ASWs provide access to ECS instances and are uplinked to ISWs.
OOB	The out-of-band switch.
OMR	The out-of-band management switch.
OASW	The out-of-band access switch. OASWs are connected to Intelligent Platform Management Interface (IPMI) cards on servers.
ACS	The advanced console server. An ACS is connected to the console port of a network device for management purposes.

Logical zones in the network architecture
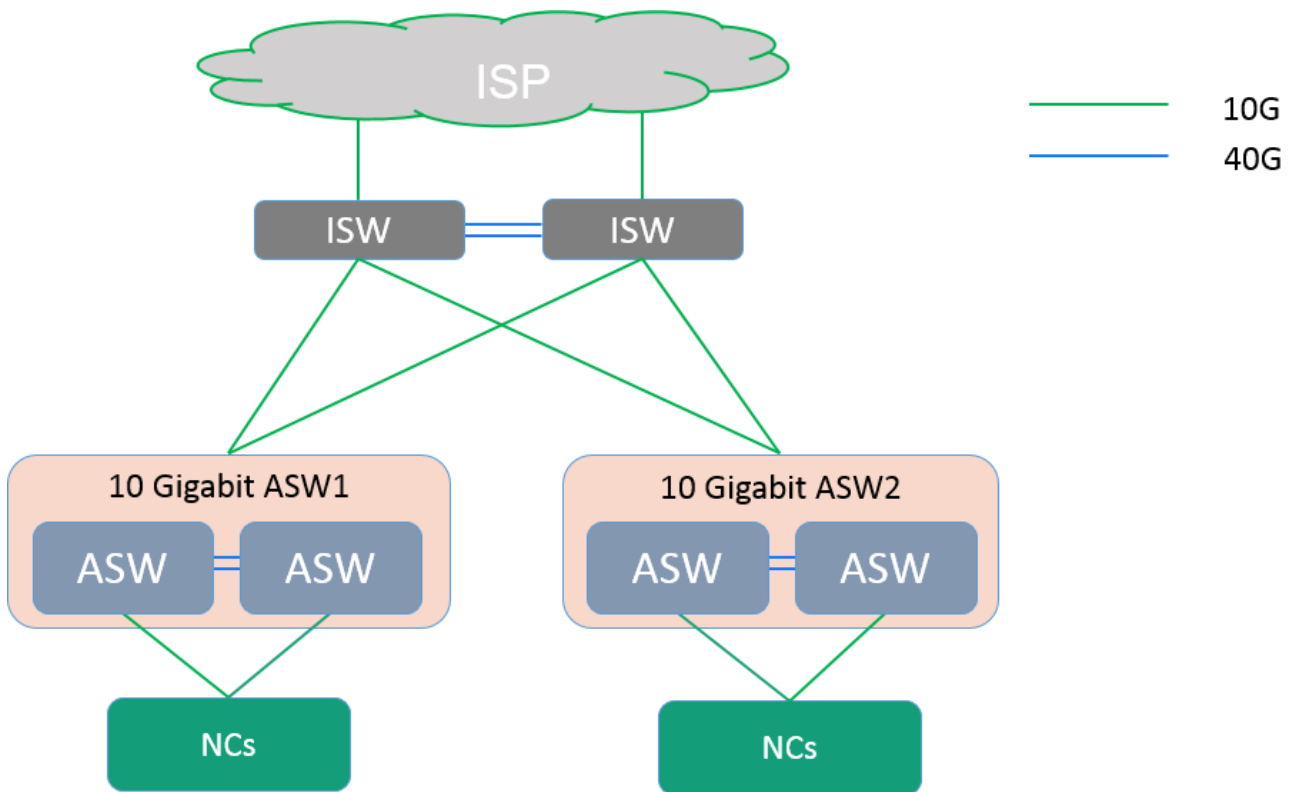
In the network architecture of Apsara Stack Agility, ISWs provide access to Internet service providers

(ISPs) and are internally connected to ASWs. The external bandwidth can be configured to suit your

needs. In this architecture, two ISWs are always deployed. The two ISWs are interconnected at a

bandwidth of 2 × 40 Gbit/s, and are downlinked to each ASW group at a bandwidth of 320 Gbit/s.

In the network architecture of Apsara Stack Agility, ASWs are connected to servers to provide network

capacity for all cloud services. Two ASWs are stacked to form a group. Apsara Stack Agility supports

up to two groups of ASWs and up to 96 servers. Each ASW group is connected to an ISW at a

bandwidth of 320 Gbit/s, and connected to a server at a bandwidth of 960 Gbit/s. The network

convergence ratio is 1:3.

All servers are configured with two network interface controllers (NICs). Each server is connected to

two ASWs by means of NIC bonding and provides 20 Gbit/s outbound bandwidth.

OOB management network
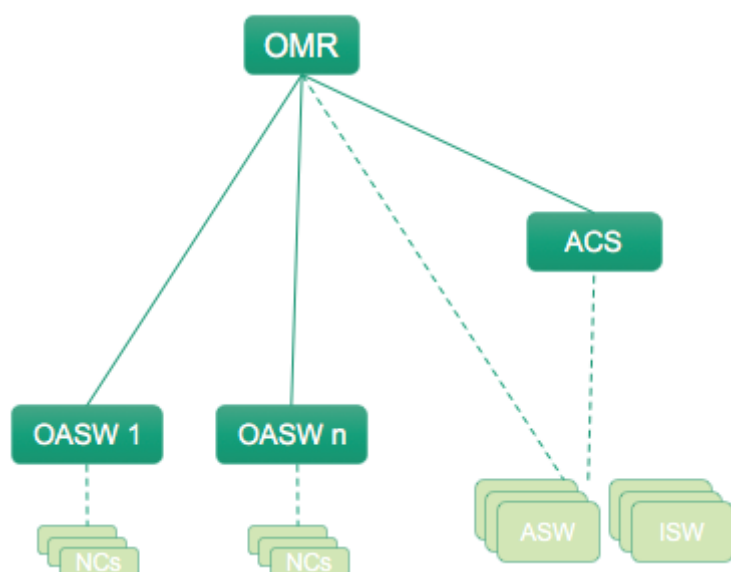
The OOB management network in the network architecture of Apsara Stack Agility manages servers

and switches in a cluster. This network is necessary for Apsara Infrastructure Management Framework

to perform operations such as installing and restarting physical servers.

- Server management network

  The IPMI port of each server is uplinked to an OASW through a GE network cable. Each OASW

  provides 48 GE network ports. The OASW supports Layer-2 passthrough and is uplinked to an

  OMR. The gateway function is configured on the OMR.

- Management port connection in the network device zone

  The management port of each network device is uplinked to an OMR through a GE network cable.

- Console port connection in the network device zone

  The console port of each network device is uplinked to an ACS through a GE network cable. The

  ACS is uplinked to an OMR.

1.3.3.2. Virtual network architecture

Container networks can be used in the virtualized deployment scenario in Rama or Nimitz mode.

Network architecture

Container networks use the VLAN solution, deliver better performance, and support on-site

interconnection. Container networks use the underlay network plane forwarding to avoid negative

impacts of tunnel encapsulation on the performance. By default, container networks use the Rama

mode. The following figure shows the architecture of a container network in Rama mode.
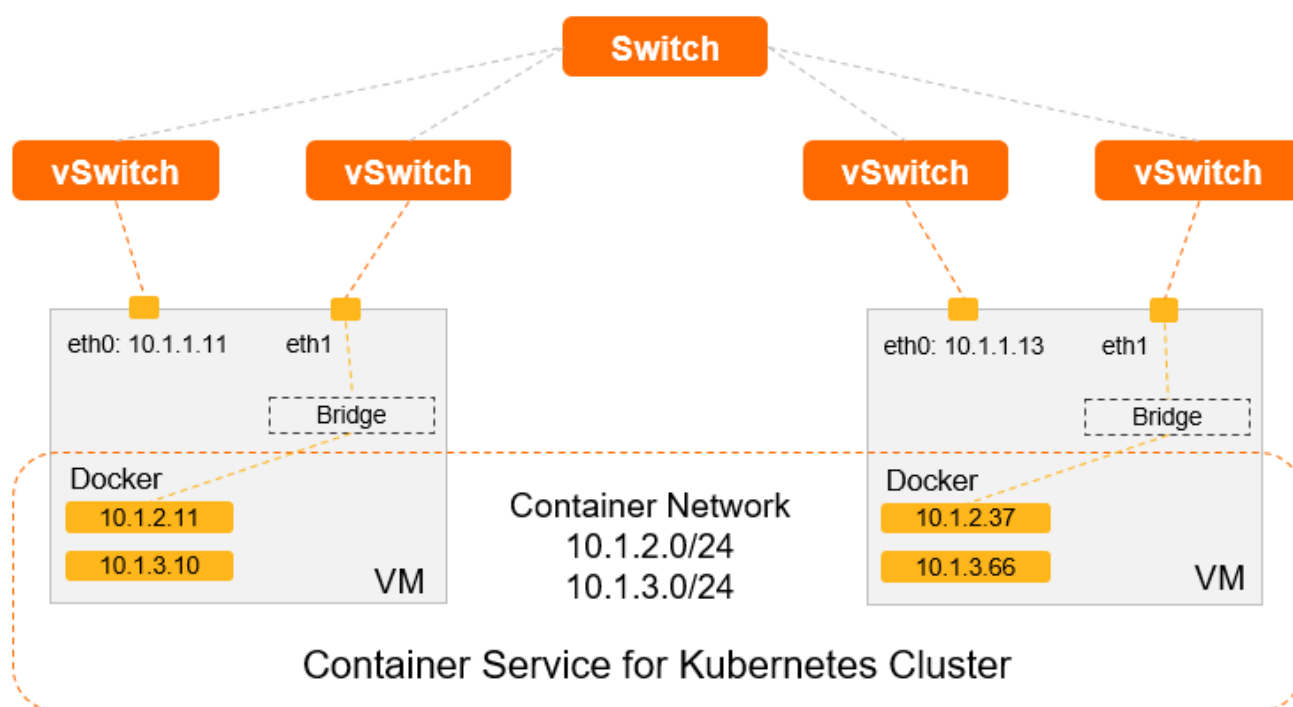
A container network has the following requirements:

- Each virtual machine (VM) is configured with two 10 GE network interface controllers (NICs). One

  NIC acts as the management NIC (eth0) of the VM and the other acts as the container network

  NIC (eth1). The VM platform ensures the network reliability.

- The management IP address of the VM is configured only on the management NIC (eth0). No IP address is configured on the container network NIC (eth1). Rama automatically creates a network bridge for eth1 and connects to the container network.

- The management networks of VMs must be interconnected with the container network.

- The VM where Bootstrap is deployed must be connected to the work of the VM where PaaS clusters are deployed.

Network mode

Container networks can be used in Rama or Nimitz mode. By default, the Rama mode is used. We also recommend that you use the Rama mode. The Rama mode supports all cloud services except specific services. The Nimitz mode supports only some cloud services. The following table describes the benefits and limits of the two modes.

Note You must provide external load balancers in Rama or Nimitz mode.

Network mode	Benefit	Limit
Rama	- An underlay network is used and the performance overheads are low. - The container and host networks are interconnected. You can connect to the	- The pod and host CIDR blocks must be interconnected. - You must provide additional IP addresses for the container network.

	container network beyond clusters. • You can deploy multiple access switches (ASWs) and CIDR blocks.	
Nimitz	• An overlay network is used. You do not need to provide additional IP addresses. The container network is able to adapt to your network. • The container network is isolated from external networks. The pod network cannot be connected from external networks. This enhances the network security.	• The performance overheads are high. Network performance losses can be up to 20%. • Packets, especially small packets, can occupy additional bandwidth. • The pod network cannot be connected from external networks. This way, some service requirements fail to be met.

1.3.4. Security architecture

Cloud products have both frontend services and backend systems. Therefore, the security architecture

of Apsara Stack Agility is divided into two layers: the platform layer and the user layer.

Apsara Stack Agility provides comprehensive security capabilities from underlying communication

protocols all the way to upper-layer applications to secure your data and access. All consoles must be

accessed using HTTPS certificates. Apsara Stack Agility provides a complete role authorization

mechanism to ensure secure and controlled access to resources in multi-tenant mode. Apsara Stack

Agility supports a variety of security roles, including security administrators, system administrators, and

security auditors.

Hierarchical security architecture of Apsara Stack Agility
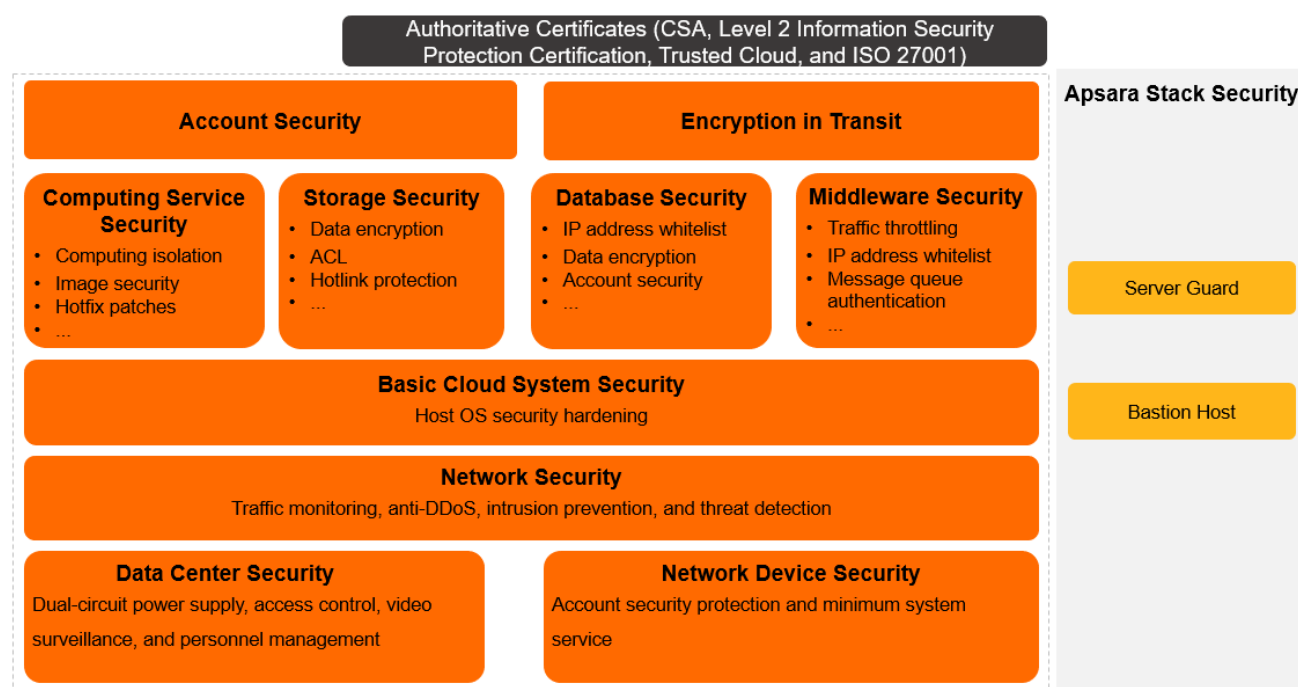
1.3.5. Base modules

Apsara Stack Agility base consists of three module types, all of which provide support for the

deployment and O&M of the cloud platform.

Base modules

Module	Function description

OPS modules	YUM	The installation package. Software repositories are deployed during the initial installation stage. These repositories are used to install the operating system and deploy application packages such as the Apsara system and dependent modules of Apsara Stack Agility on hosts.
	Clone	The virtual machine (VM) cloning service.
	NTP	The clock source service. NTP is deployed on Apsara Stack Agility hosts to synchronize time from the standard NTP clock source to other hosts.
	DNS	The domain name resolution service. DNS provides forward and reverse resolution of domain names for the internal Apsara Stack Agility environment. DNS runs a bind instance on each of the two OPS machines and uses keepalived to provide high availability services. If one machine

Base middleware		fails, the other machine automatically takes over its work.
	Dubbo	The distributed remote procedure call (RPC) service.
	Tair	The caching service.
	MQ	The message queuing service.
	ZooKeeper	The distributed coordination service.
	Diamond	The configuration management service.
Basic modules of the base	Apsara Infrastructure Management Framework	The data center management system.
	Monitoring System	The data center monitoring system.
	Metadatabase	The metadatabase.
	POP	The Apsara Stack OpenAPI platform.
	OAM	The account system.

RAM	The authentication and authorization system.
WebApps	The service that provides support for the Apsara Stack Operations (ASO) console.
KubeMaster	The Kubernetes control node of Apsara Stack Agility PaaS.
KubeWorker	The Kubernetes worker node of Apsara Stack Agility PaaS.

1.4. Product panorama

Apsara Stack Agility offers a wide range of services to meet the diverse needs of different users.

Basic services

Apsara Stack Agility provides basic computing, network, and storage capabilities. The main services

include Elastic Compute Service (ECS), Server Load Balancer (SLB), and Virtual Private Cloud (VPC).

Storage services

The storage services include Object Storage Service (OSS).

Database services

Apsara Stack Agility provides a variety of database engines that can communicate with each other. The

main services include ApsaraDB RDS for MySQL, PolarDB-X, and Data Transmission Service (DTS).

1.5. Scenarios

Apsara Stack Agility provides flexible and scalable industrial solutions for customers of different scales

and sectors. Apsara Stack Agility can create customized solutions based on the business traits of

different sectors such as industry, agriculture, transportation, government, finance, and education to

provide users with end-to-end products and services.

Scenarios for small-scale private cloud that focuses on IaaS

Apsara Stack Agility delivers the same IaaS experience as Alibaba Cloud public cloud, such as

providing virtual machines (VMs), virtual networks, load balancing capabilities, and disks.

Scenarios for small-scale private cloud that focuses on storage service

The small-scale private cloud that focuses on storage service targets traditional storage markets and is

characterized by high availability, high throughput, low latency, and high scalability. The storage

services include Object Storage Service (OSS).

Values and features

- The integrated storage platform improves resource utilization and greatly reduces deployment and

  operations costs.

- The total bandwidth increases linearly with the expansion of nodes, and system performance is

  guaranteed during elastic scaling to adapt to future business trends.

- Compared with traditional storage, the small-scale private cloud that focuses on storage service

  greatly improves the concurrent processing capability and read/write speed.

More and more enterprises want to build distributed databases to support Internet-based businesses. These enterprises include financial institutions such as banks, securities and insurance firms, and fund companies.

Scenarios for small-scale private cloud that focuses on database service

The small-scale private cloud that focuses on database service targets traditional database markets and provides high-availability and high-performance transactional or analytic database services. The main services include ApsaraDB RDS for MySQL, PolarDB-X, and Data Transmission Service (DTS).

Values and features

- The integrated database platform improves resource utilization and greatly reduces deployment and operations costs.

- The total bandwidth increases linearly with the expansion of nodes, and system performance is guaranteed during elastic scaling to adapt to future business trends.

- ApsaraDB for RDS Enterprise Edition offers strong consistency.

- Failure of any single server does not affect services.

- The entire data center can be automatically restored after a power outage or network disconnection.

More and more enterprises want to build databases. These enterprises include financial institutions such as banks, securities and insurance firms, and fund companies. This platform is ideal for these scenarios.

2.Object Storage Service (OSS)

2.1. What is OSS?

Object Storage Service (OSS) is a secure, cost-effective, and highly reliable cloud storage service

provided by Alibaba Cloud. It enables you to store a large amount of data in the cloud.

OSS is an immediately available storage solution that has unlimited storage capacity. Compared with

user-created server storage, OSS has outstanding advantages in reliability, security, cost-effectiveness,

and data processing capabilities. OSS enables you to store and retrieve a variety of unstructured data

objects, such as texts, images, audios, and videos over the network at any time.

OSS is an object storage service based on key-value pairs. Files uploaded to OSS are stored as

objects in buckets. You can obtain the content of an object based on the object key.

In OSS, you can:

- Create a bucket and upload objects to the bucket.

- Obtain an object URL from OSS to share or download the object.

- Modify the attributes or metadata of a bucket or an object, and configure ACL for the bucket or the

  object.

- Perform basic and advanced operations in the OSS console.

- Perform basic and advanced operations by using SDKs or calling RESTful API operations in your

  application.

2.2. Benefits

OSS provides secure, cost-effective, and highly reliable services for storing large amounts of data in the

cloud. This topic compares OSS with the traditional user-created server storage to show the benefits of

OSS.

Advantages of OSS over user-created server storage

Item	OSS	User-created server storage
Reliability	Automatically stores multiple copies of data for backup.	• Prone to errors due to low hardware reliability. If a disk has a bad sector, data may be lost. • Manual data restoration is complex and requires a lot of time and technical resources.
Security	• Provides hierarchical security protection for enterprises. • Provides resource isolation mechanisms for multiple tenants and supports zone-disaster recovery.	• Additional scrubbing devices and black hole policy-related services are required. • A separate security mechanism is required.

	• Provides various authentication and authorization mechanisms, as well as features such as whitelists, hotlink protection, RAM, and Security Token Service (STS) for temporary access.	
Data processing	Provides Image Processing (IMG).	Equipment for data processing must be purchased and deployed separately.

More benefits of OSS

- Ease of use

  Provides standard RESTful API operations (some compatible with Amazon S3 API operations), a wide range of SDKs, client tools, and console. You can upload, download, retrieve, and manage large amounts of data for websites or mobile applications the way you use regular file systems.

  o The number and size of objects are not limited.

  o Streaming writes and reads are supported, which is suitable for business scenarios where you must simultaneously read and write videos and other large objects.

  o Lifecycle management is supported. You can delete expired data in batches.

- Powerful and flexible security mechanisms

  Flexible authentication and authorization mechanisms are available. OSS provides STS and URL-based authentication and authorization mechanisms, whitelists, hotlink protection, and RAM.

- Rich image processing functions

  Supports format conversion, thumbnails, cropping, watermarking, resizing for objects in formats such as JPG, PNG, BMP, GIF, WebP, and TIFF.

2.3. Architecture

OSS is a storage solution that is built on the Apsara system. It is based on the infrastructure such as Apsara Distributed File System and SchedulerX. The infrastructure provides OSS and other Alibaba Cloud services with importance features such as distributed scheduling, high-speed networks, and distributed storage. The following figure shows the OSS architecture.

OSS architecture
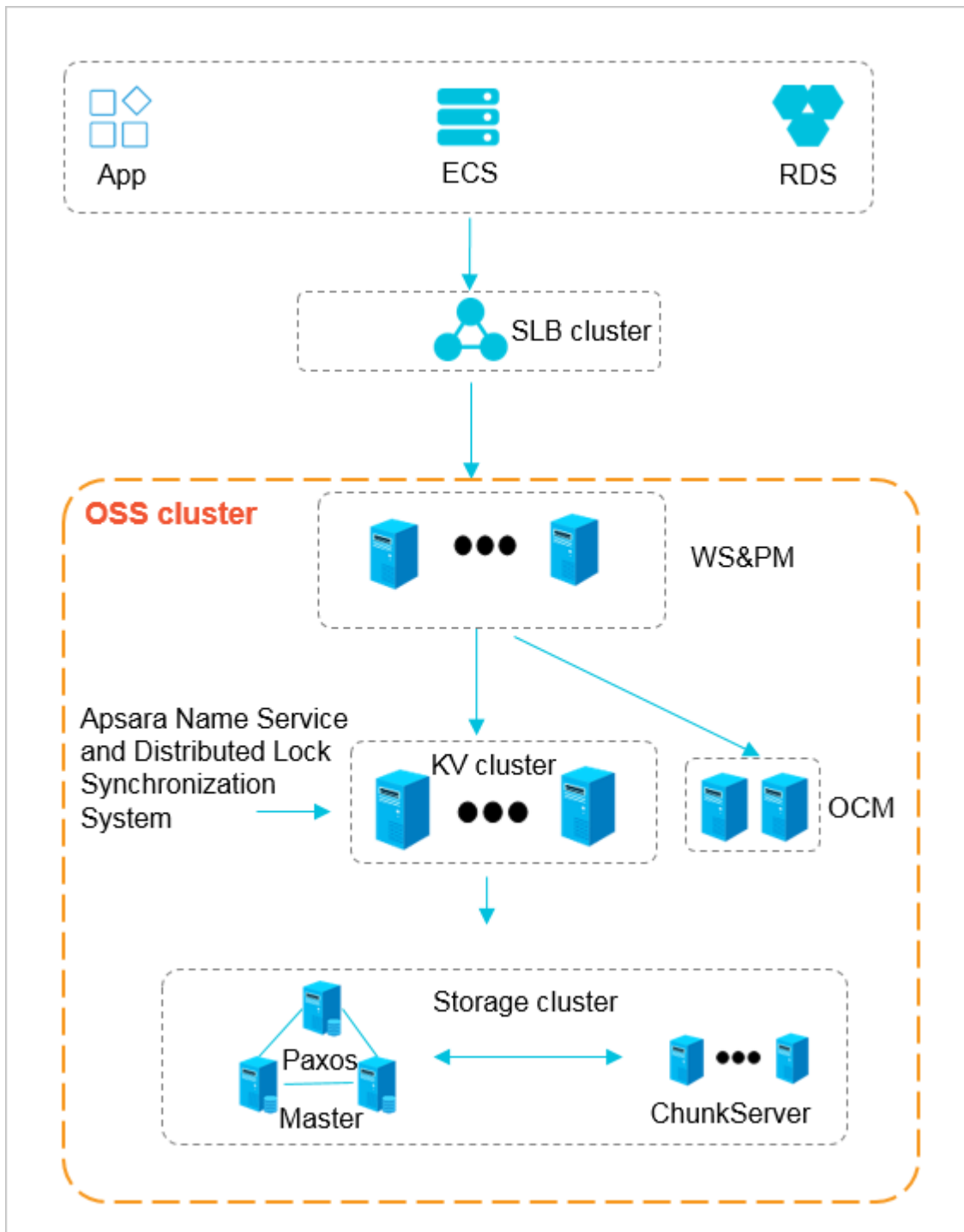
- WS & PM: the protocol layer that receives and authenticates the request sent by using a RESTful

  protocol. If the authentication is successful, the request is forwarded to KVEngine for further

  processing. If the authentication fails, an error message is returned.

- KV cluster: used to process structured data, including reading and writing data based on object names. The KV cluster also supports sporadic bursts of requests. When a service has to run on a different physical server due to a change to the service coordination cluster, the KV cluster can coordinate and find the access point.

- Storage cluster: Metadata is stored in the master node. A distributed message consistency protocol of Paxos is adopted between Master nodes to ensure the consistency of metadata. This method ensures efficient distributed storage of and access to objects.

2.4. Features

This topic lists the common features of OSS.

Before you start to use OSS, we recommend that you have a good understanding of basic terms used in OSS, such as bucket, object, region, and endpoint.

The following table describes features of OSS.

OSS features

Category	Feature	Description
Bucket	Create buckets	Before you upload an object to OSS, you must create a bucket to store the object.
	Delete buckets	If you no longer use a bucket, delete it to avoid further fees.

	Modify bucket ACL	OSS supports ACL for access control. You can configure the ACL of a bucket when you create the bucket or modify the ACL of a created bucket.
	Configure static website hosting	You can configure static website hosting for your bucket and access this static website through the bucket domain name.
	Configure hotlink protection	To prevent additional fees caused by unauthorized access to the data in your bucket, you can configure hotlink protection for your buckets based on the Referer field in HTTP requests.
	Manage CORS	OSS provides cross-origin resource sharing (CORS) over HTML5 to implement cross-origin access.
	Configure lifecycle rules	You can define and manage lifecycle rules for all or a subset of objects in a bucket. You can configure lifecycle rules to manage multiple objects and automatically delete parts.
Object	Upload objects	You can upload any type of objects to a bucket.

	Create folders	You can manage OSS folders the way you manage folders in Windows.
	Search for objects	You can search for objects whose names contain the same prefix in a bucket or folder.
	Obtain object URLs	You can obtain the URL of an object to share or download the object.
	Delete objects	You can delete a single object or multiple objects.
	Delete folders	You can delete a single folder or multiple folders.
	Modify object ACL	You can configure the ACL of an object when you upload it or modify the ACL of an uploaded bucket.
	Manage parts	You can delete all or some parts from a bucket.
Image processing	IMG	You can perform operations such as format conversion, cropping, scaling, rotating, watermarking, style encapsulation on images stored in OSS.
Access control for VPC	Single Tunnel	You can create single tunnels to access OSS resources through VPC.

API	API	OSS supports RESTful API operations and provides examples.
SDK	SDK	OSS supports development based on SDKs for various programming languages and provides examples.

2.5. Scenarios

This topic describes the application scenarios of OSS.

Massive storage for image, audio, and video applications

OSS can be used to store large amounts of data, such as images, audio and video data, and logs. OSS supports various devices. Websites and mobile applications can directly read or write OSS data. OSS supports file writing and streaming writing.

Dynamic and static content separation for websites and mobile applications

By using the BGP bandwidth, you can download data from OSS with an ultra-low latency.

Offline data storage

OSS provides storage with low cost and high availability. Therefore, you can use OSS to store enterprise data that needs to be archived offline for a long period.

2.6. Limits

This topic describes the limits and performance metrics of OSS.

Item	Limit
Bucket	• You can create a maximum of 100 buckets. • After a bucket is created, its name and region cannot be modified.
Object upload	• Objects larger than 5 GB cannot be uploaded by using the following modes: console upload, simple upload, form upload, or append upload. To upload an object that is larger than 5 GB, you must use multipart upload. The size of an object uploaded by using multipart upload cannot exceed 48.8 TB. • If you upload an object that has the same name of an existing object in OSS, the new object will overwrite the existing object.
Object deletion	• Deleted objects cannot be recovered. • You can delete up to 100 objects at a time in the OSS console. To delete more than 100 objects at a time, you must call an API operation or use an SDK.
Lifecycle	You can configure up to 1,000 lifecycle rules for each bucket.

2.7. Terms

This topic describes several basic terms used in OSS.

Object

The basic unit for data operations in OSS. Objects are also known as OSS files. An object is composed

of object metadata, object content, and a key. A key can uniquely identify an object in a bucket. Object

metadata is a group of key-value pairs that define the properties of an object, such as the last

modification time and the object size. You can also assign user metadata to the object.

The lifecycle of an object starts when the object is uploaded, and ends when it is deleted. During the

lifecycle, the object cannot be modified. OSS does not support modifying objects. If you want to modify

an object, you must upload a new object with the same name as the existing object to replace it.

Note Unless otherwise stated, objects and files mentioned in OSS documents are collectively called

objects.

Bucket

A container for OSS objects. Each object in OSS is contained in a bucket. You can configure and

modify the attributes of a bucket to manage ACLs and lifecycle rules of the bucket. These attributes

apply to all objects in the bucket. Therefore, you can create different buckets to meet different

management requirements.

- OSS does not use a hierarchical structure for objects, but instead uses a flat structure. All

  elements are stored as objects in buckets. However, OSS supports folders as a concept to group

  objects and simplify management.

- You can create multiple buckets.

- A bucket name must be globally unique within OSS. Bucket names cannot be changed after the buckets are created.

- A bucket can contain an unlimited number of objects.

Strong consistency

A feature requires that object operations in OSS be atomic, which indicates that operations can only either succeed or fail. There are no intermediate states. To ensure that users can access only complete data, OSS does not return corrupted or partial data.

Object-related operations in OSS are highly consistent. For example, when a user receives an upload (PUT) success response, the uploaded object can be read immediately, and copies of the object have been written to multiple devices for redundancy. Therefore, there are no situations where data is not obtained when you perform the read-after-write operation. The same is true for delete operations. After you delete an object, the object and its copies no longer exist.

Similar to traditional storage devices, modifications are immediately visible in OSS while consistency is guaranteed.

Comparison between OSS and file systems

OSS is a distributed object storage service that stores objects based on key-value pairs. You can retrieve object content based on unique object keys. For example, object name *test1/test.jpg* does not necessarily indicate that the object is stored in a directory named test1. In OSS, *test1/test.jpg* is only a

string. There is nothing essentially different between test1/test.jpg and *a.jpg*. Therefore, similar amounts of resources are consumed regardless of which object you access.

A file system uses a typical tree index structure. To access a file named *test1/test.jpg*, you must first access the test1 directory and then search for the *test.jpg* file in this directory. This makes it easy for a file system to support folder operations, such as renaming, deleting, and moving directories because these operations are only performed on directories. However, the performance of a file system depends on the capacity of a single device. The more files and directories that are created in the file system, the more resources and time are consumed.

You can simulate similar folder functions of a file system in OSS, but such operations are costly. For example, if you want to rename the test1 directory as test2, OSS must copy all objects whose names start with test1/ to generate objects whose names start with test2. This operation consumes a large amount of resources. Therefore, we recommend that you do not perform such operations in OSS.

Objects stored in OSS cannot be modified. A specific API operation must be called to append an object, and the generated object is different from objects uploaded by using other methods. To modify even a single byte, you must upload the entire object again. A file system allows you to modify files. You can modify the content at a specified offset location or truncate the end of a file. These features make file systems suitable for more general scenarios. However, OSS supports a large amount of concurrent access, whereas the performance of a file system is subject to the performance of a single device.

We recommend that you do not map operations on OSS objects to file systems because it is inefficient. If you attach OSS as a file system, we recommend that you only add new files, delete files, and read

files. You can make full use of OSS advantages, such as the capability to process and store large

amounts of unstructured data such as images, videos, and documents.

3.ApsaraDB for RDS

3.1. What is ApsaraDB for RDS?

ApsaraDB for RDS is a stable, reliable, and automatically scaling online database service. Based on the distributed file system and high-performance storage, ApsaraDB for RDS allows you to easily perform database operations and maintenance with its set of solutions for disaster recovery, backup, restoration, monitoring, and migration.

Originally based on a branch of MySQL, ApsaraDB RDS for MySQL has proven its performance and throughput during the high-volume concurrent traffic of Double 11. ApsaraDB RDS for MySQL provides whitelist configuration, backup and restoration, transparent data encryption, data migration, and management for instances, accounts, and databases. It also provides the following advanced features:

- Read-only instance: In scenarios where RDS has a small number of write requests but a large number of read requests, you can enable read/write splitting to distribute read requests away from the primary instance. Read-only instances allow ApsaraDB RDS for MySQL 5.6 to automatically scale the reading capability and increase the application throughput when a large amount of data is being read.

- Data compression: ApsaraDB RDS for MySQL 5.6 allows you to compress data by using the TokuDB storage engine. Data transferred from the InnoDB storage engine to the TokuDB storage engine can be reduced by 80% to 90% in volume. 2 TB of data in InnoDB can be compressed to 400 GB or less in TokuDB. In addition to data compression, TokuDB supports transaction and online DDL operations. TokuDB is compatible with MyISAM and InnoDB applications.

3.2. Benefits

3.2.1. Ease of use

ApsaraDB for RDS is a ready-to-use service featuring on-demand upgrades, convenient management, high transparency, and high compatibility.

Ready-to-use

You can use the API to create instances of any specified RDS instance type.

On-demand upgrade

When the database load or data storage capacity changes, you can upgrade the RDS instance by changing its type. The upgrades do not interrupt the data link service.

Transparency and compatibility

ApsaraDB for RDS is used in the same way as the native RDS database engine, allowing it to be adopted easily without the need to learn new database engines. ApsaraDB for RDS is compatible with existing programs and tools. Data can be migrated to ApsaraDB for RDS through ordinary import and export tools.

Easy management

Alibaba Cloud is responsible for the routine maintenance and management tasks for ApsaraDB for RDS such as troubleshooting hardware and software issues or issuing database patches and updates. You can also manually add, delete, restart, back up, and restore databases through the Apsara Stack console.

3.2.2. High performance

ApsaraDB for RDS implements parameter optimization, SQL optimization, and high-end backend

hardware to achieve high performance.

Parameter optimization

All RDS instance parameters have been optimized over their several years of production. Professional

database administrators continue to optimize RDS instances over their lifecycles to ensure that

ApsaraDB for RDS runs at peak efficiency.

SQL optimization

ApsaraDB for RDS locks inefficient SQL statements and provides recommendations to optimize code.

High-end backend hardware

All servers used by ApsaraDB for RDS are evaluated by multiple parties to ensure stability.

3.2.3. High security

ApsaraDB for RDS implements anti-DDoS protection, access control, system security, and transparent

data encryption (TDE) to guarantee the security of your databases.

DDoS attack prevention

Note You must activate Alibaba Cloud security services to use this feature.

When you access an ApsaraDB for RDS instance from the Internet, the instance is vulnerable to DDoS

attacks. When a DDoS attack is detected, the RDS security system first scrubs inbound traffic. If traffic

scrubbing is insufficient or if the black hole threshold is reached, black hole filtering is triggered.

Triggering conditions for traffic scrubbing and black hole filtering are listed as follows:

- Traffic scrubbing:

  Traffic scrubbing only targets traffic from the Internet and does not affect normal operations of

  your instance.

  ApsaraDB for RDS triggers and stops traffic scrubbing automatically. Traffic scrubbing is triggered

  for a single ApsaraDB for RDS instance if any of the following conditions are met:

  o Packets per second (PPS) reaches 30,000.

  o Bits per second (BPS) reaches 180 Mbit/s.

  o The number of new concurrent connections per second reaches 10,000.

  o The number of active concurrent connections reaches 10,000.

  o The number of inactive concurrent connections reaches 10,000.

- Black hole filtering:

  Black hole filtering only targets traffic from the Internet. If an RDS instance is undergoing black

  hole filtering, the instance cannot be accessed from the Internet and connected applications will

  not be available. Black hole filtering guarantees availability of RDS.

  Conditions for triggering black hole filtering are listed as follows:

  o BPS reaches 2 Gbit/s.

  o Traffic scrubbing is ineffective.

  Black hole filtering is automatically stopped 2.5 hours after being triggered.

Access control

You can configure an IP address whitelist for ApsaraDB for RDS to allow access for specified IP addresses and deny access for all others.

Each account can only view and operate their own respective database.

System security

ApsaraDB for RDS is protected by several layers of firewalls capable of blocking a variety of attacks to secure data.

ApsaraDB for RDS servers cannot be logged onto directly. Only the ports required for specific database services are provided.

ApsaraDB for RDS servers cannot initiate an external connection. They can only receive access requests.

3.2.4. High reliability

ApsaraDB for RDS provides hot standby, multi-copy redundancy, data backup, and data recovery to achieve high reliability.

Hot standby

ApsaraDB for RDS adopts a hot standby architecture. If the primary server fails, services will fail over to the secondary server within seconds. Applications running on the servers are not affected by the failover process and will continue to run normally.

Multi-copy redundancy

ApsaraDB for RDS servers implement a RAID architecture to store data. Data backup files are stored

on OSS.

Data backup

ApsaraDB for RDS provides an automatic backup mechanism. You can schedule backups to be

performed periodically, or manually initiate temporary backups as necessary to meet your business
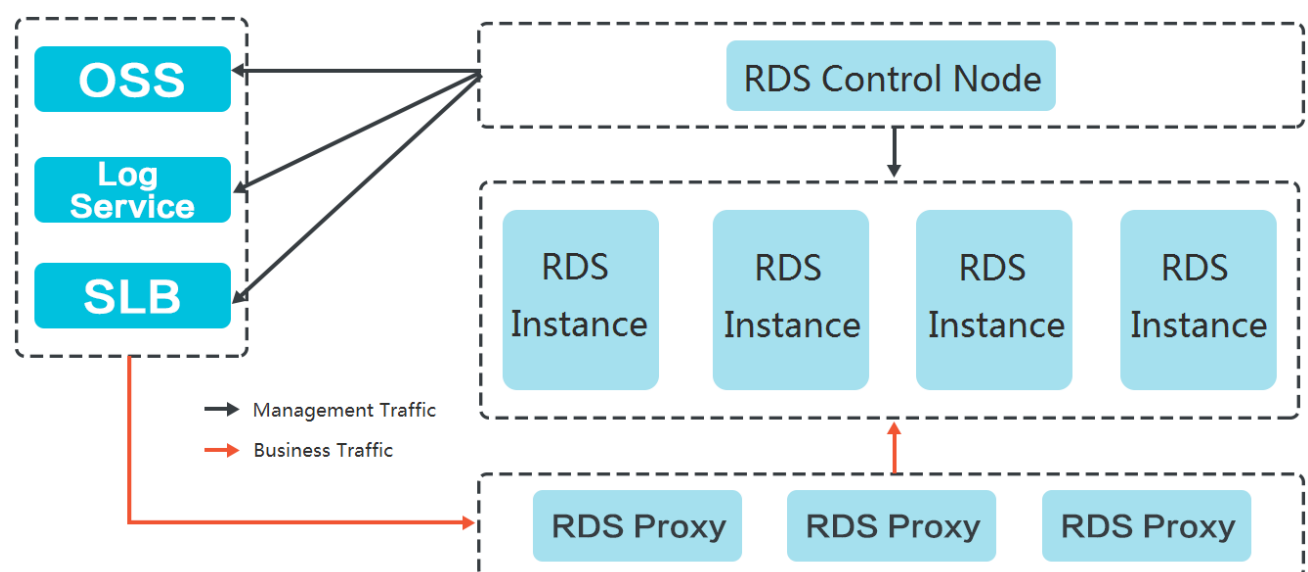
needs.

Data recovery

Data can be restored from backup sets or cloned instances created at previous points in time. After

data is verified, the data can be migrated back to the primary RDS instance.

3.3. Architecture

The following figure shows the system architecture of ApsaraDB for RDS.

RDS system architecture

3.4. Features

3.4.1. Data link service

ApsaraDB for RDS provides all data link services, including DNS, Server Load Balancer (SLB), and

Proxy.

ApsaraDB for RDS uses native database engines with similar database operations to minimize learning

costs and facilitate database access.

DNS

The DNS module can dynamically resolve domain names to IP addresses. Therefore, IP address

changes do not affect the performance of RDS instances. After the domain name of an RDS instance is

configured in the connection pool, the RDS instance can be accessed even if its corresponding IP

address changes.

For example, the domain name of an ApsaraDB for RDS instance is test.rds.aliyun.com, and its

corresponding IP address is 10.10.10.1. The instance can be accessed when either test.rds.aliyun.com

or 10.10.10.1 is configured in the connection pool of a program.

After a zone migration or version upgrade is performed for this ApsaraDB for RDS instance, the IP

address may change to 10.10.10.2. If the domain name test.rds.aliyun.com is configured in the

connection pool, the instance can still be accessed. However, if the IP address configured in the

connection pool is 10.10.10.1, the instance will no longer be accessible.

SLB

The SLB module provides both the internal IP address and public IP address of an ApsaraDB for RDS

instance. Therefore, server changes do not affect the performance of the instance.

For example, the internal IP address of an RDS instance is 10.1.1.1, and the corresponding Proxy or

DB Engine runs on 192.168.0.1. The SLB module typically redirects all traffic destined for 10.1.1.1 to

192.168.0.1. If 192.168.0.1 fails, another server in hot standby status with the IP address 192.168.0.2

will take over for the initial server. In this case, the SLB module will redirect all traffic destined for

10.1.1.1 to 192.168.0.2, and the RDS instance will continue to provide services normally.

Proxy

The Proxy module provides a number of features including data routing, traffic detection, and session

persistence.

- Data routing: aggregates the distributed complex queries found in big data scenarios and provides

  the corresponding capacity management capabilities.

- Traffic detection: reduces SQL injection risks and supports SQL log backtracking when necessary.

- Session persistence: prevents database connection interruptions when faults occur.

3.4.2. High-availability service

The high-availability (HA) service consists of modules such as the Detection, Repair, and Notice.

The HA service guarantees the availability of data link services and processes internal database

exceptions.

Detection

The Detection module checks whether the primary and secondary nodes of the DB   Engine are

providing their services normally. The HA node uses heartbeat information taken at 8 to 10 second

intervals to determine the health status of the primary node. This information, along with the health

status of the secondary node and heartbeat information from other HA nodes, provides a reference for the Detection module. All this information helps the module avoid misjudgment caused by exceptions such as network jitter. Failover can be completed within 30 seconds.

Repair

The Repair module maintains the replication relationship between the primary and secondary nodes of the DB Engine. It can also correct errors that occur on either node during normal operations.

For example:

- It can automatically restore primary/secondary replication after a disconnection.

- It can automatically repair table-level damage to the primary or secondary node.

- It can save and automatically repair the primary or secondary node in case of crashes.

Notice

The Notice module informs the SLB or Proxy module of status changes to the primary and secondary nodes to ensure that you always access the correct node.

For example, the Detection module discovers problems with the primary node and instructs the Repair module to resolve these problems. If the Repair module fails to resolve a problem, it instructs the Notice module to perform traffic switchover. The Notice module forwards the switching request to the SLB or Proxy module, and then all traffic is redirected to the secondary node. Meanwhile, the Repair module creates a new secondary node on a different physical server and synchronizes this change back to the Detection module. The Detection module rechecks the health status of the instance.

HA policies

Each HA policy defines a combination of service priorities and data replication modes defined to meet the needs of your business.

There are two service priorities:

- Recovery time objective (RTO): The database preferentially restores services to maximize the availability time. Use the RTO policy if you require longer database uptime.

- Recovery point objective (RPO): The database preferentially ensures data reliability to minimize data loss. Use the RPO policy if you require high data consistency.

There are three data replication modes:

- Asynchronous replication (Async): When an application initiates an update request such as add, delete, or modify operations, the primary node responds to the application immediately after the primary node completes the operation. The primary node then replicates data to the secondary node asynchronously. This means that the operation of the primary database is not affected if the secondary node is unavailable. Data inconsistencies may occur if the primary node is unavailable.

- Forced synchronous replication (Sync): When an application initiates an update request such as add, delete, or modify operations, the primary node replicates data to the secondary node immediately after the primary node completes the operation. The primary node then waits for the secondary node to return a success message before the primary node responds to the application. The primary node replicates data to the secondary node synchronously. Unavailability of the secondary node will affect the operation on the primary node. Data will remain consistent even when the primary node is unavailable.

- Semi-synchronous replication (Semi-Sync): Data is typically replicated in Sync mode. When trying to replicate data to the secondary node, if an exception occurs causing the primary and secondary nodes to be unable to communicate with each other, the primary node will suspend response to the application. If the connection cannot be restored, the primary node will degrade to Async mode and restore response to the application after the Sync replication times out. In a situation such as this, the primary node becoming unavailable will lead to data inconsistency. After the secondary node or network connection is recovered, data replication between the two nodes is resumed, and the data replication mode will change from Async to Sync.

You can select different combinations of service priorities and data replication modes to improve availability based on the business features.

3.4.3. Backup and recovery service

This service supports data backup, storage, and recovery functions.

ApsaraDB for RDS can back up databases at any time and restore them to any point in time based on the backup policy, making the data more traceable.

Backup

The Backup module compresses and uploads data and logs on both the primary and secondary nodes. ApsaraDB for RDS uploads backup files to OSS and stores the backup files to a more cost-effective and persistent Archive Storage system. When the secondary node is operating properly, backup is always initiated on the secondary node. This will not affect the services on the primary node. When the secondary node is unavailable or damaged, the Backup module initiates backup on the primary node.

Recovery

The Recovery module restores backup files stored on OSS to a destination node.

- Primary node rollback: rolls back the primary node to a specified point in time when an operation

  error occurs.

- Secondary node repair: creates a new secondary node to reduce risks when an irreparable fault

  occurs on the secondary node.

- Read-only instance creation: creates a read-only instance from backup files.

Storage

The Storage module uploads, stores, and downloads backup files. All backup data is uploaded to OSS

for storage. You can obtain temporary links to download backups as necessary. In certain scenarios,

the Storage module allows you to store backup files from OSS to Archive Storage for more cost-

effective and longer-term offline storage.

3.4.4. Monitoring service

ApsaraDB for RDS provides multilevel monitoring services across the physical, network, and

application layers to ensure service availability.

Service

The Service module tracks the status of services. For example, the Service module monitors whether

SLB, OSS, and other cloud services on which RDS depends are operating normally. The monitored

metrics include functionality and response time. The Service module also uses logs to determine

whether the internal RDS services are operating properly.

Network

The Network module tracks statuses at the network layer. The module monitors the connectivity between ECS and RDS and between physical RDS servers, as well as the rates of packet loss on VRouters and VSwitches.

OS

The OS module tracks the statuses of hardware and OS kernel. The monitored metrics include:

- Hardware maintenance: The OS module constantly checks the operating status of the CPU, memory, motherboard, and storage device. It can predict faults in advance and automatically submit repair reports when it determines a fault is likely to occur.

- OS kernel monitoring: The OS module tracks all database calls and analyzes the causes of slow calls or call errors based on the kernel status.

Instance

The Instance module collects the following information about ApsaraDB for RDS instances:

- Instance availability information

- Instance capacity and performance metrics

- Instance SQL execution records

3.4.5. Scheduling service

The Resource module implements the scheduling of resources and services.

Resource

The Resource module allocates and integrates underlying RDS resources when you enable and migrate instances. When you use the RDS console or an API operation to create an instance, the Resource module calculates the most suitable host to carry the traffic to and from the instance. This module also allocates and integrates the underlying resources required to migrate RDS instances. After repeated instance creation, deletion, and migration operations, the Resource module calculates the degree of resource fragmentation. It also regularly integrates resources to improve the service carrying capacity.

3.4.6. Migration service

RDS provides Data Transmission Service (DTS) to help you migrate databases quickly.

The migration service helps you migrate data from the on-premises database to ApsaraDB for RDS, or migrate data from an instance to another instance in ApsaraDB for RDS.

DTS

DTS enables data migration from on-premises databases to RDS instances or between different RDS instances.

DTS provides three migration methods: schema migration, full migration, and incremental migration.

- Schema migration

  DTS migrates the schema definitions of migration objects to the destination instance. Tables, views, triggers, stored procedures, and stored functions can be migrated in this mode.

- Full migration

  DTS migrates all data of migration objects from the source database to the destination instance.

Notice To ensure data consistency, non-transaction tables that do not have primary keys will be locked when performing a full migration. Locked tables cannot be written to. The lock duration depends on the amount of data in the tables. The tables will be unlocked only after they are fully migrated.

- Incremental migration

DTS synchronizes data changes made in the migration process to the destination instance.

Notice If a DDL operation is performed during data migration, schema changes will not be synchronized to the destination instance.
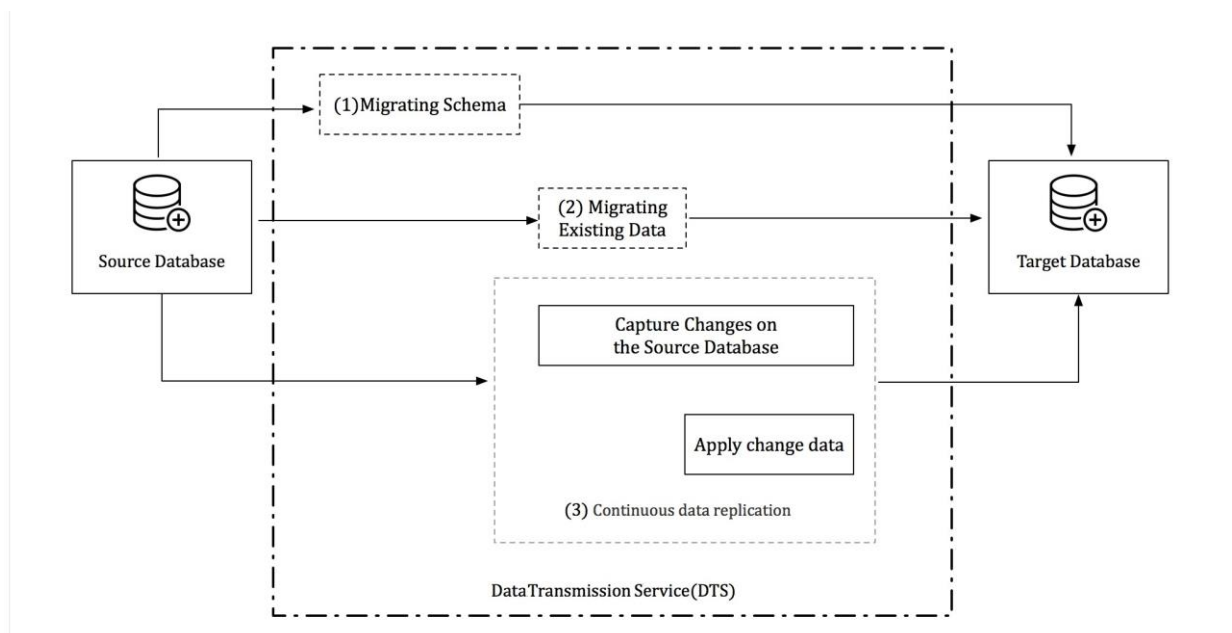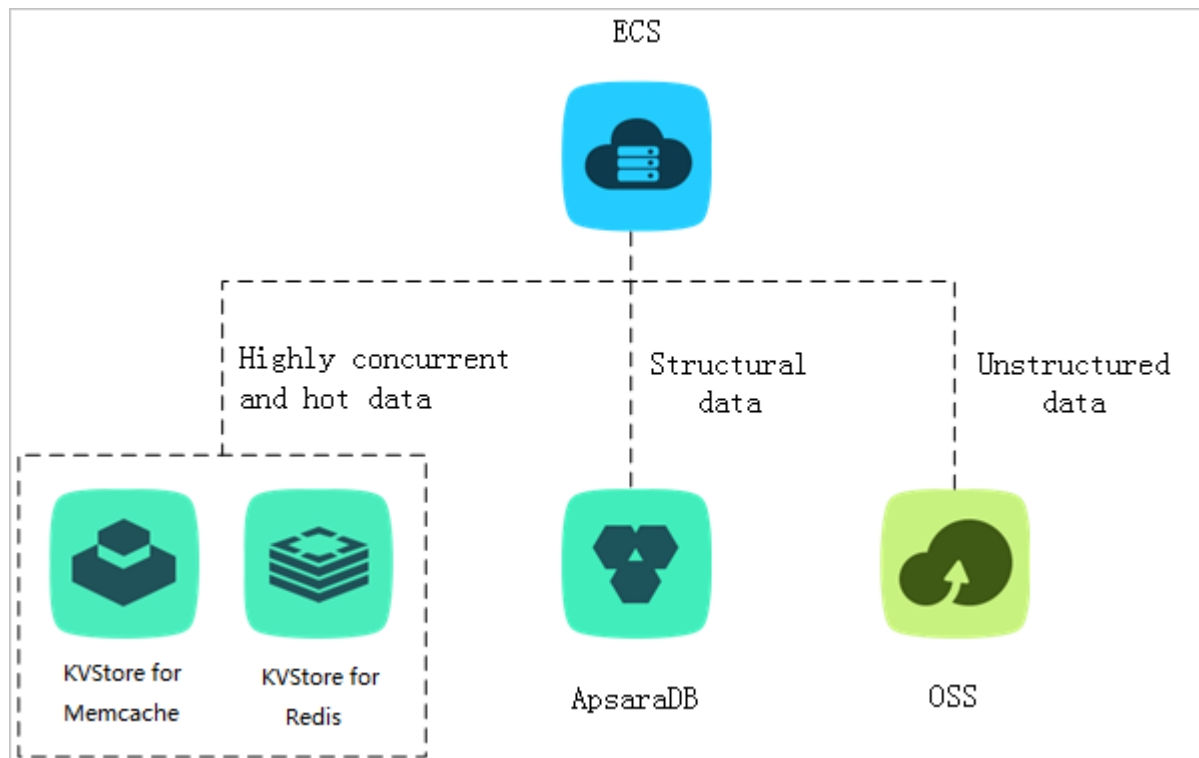

3.5. Scenarios

3.5.1. Diversified data storage

ApsaraDB for RDS provides cache data persistence and multi-structure data storage.

You can diversify the storage capabilities of ApsaraDB for RDS through services such as KVStore for Memcache, KVStore for Redis, and OSS, as shown in Diversified data storage.

Diversified data storage

Cache data persistence

ApsaraDB for RDS can be used with KVStore for Memcache and KVStore for Redis to form a high-throughput and low-latency storage solution. These cache services have the following benefits over ApsaraDB for RDS:

- High response speed: The request latency of KVStore for Memcache and KVStore for Redis is usually within just a few milliseconds.

- The cache area supports a higher number of queries per second (QPS) than ApsaraDB for RDS.

Multi-structure data storage

OSS is a secure, reliable, low-cost, and high-capacity storage service from Alibaba Cloud. ApsaraDB for RDS can be used with OSS to implement a multi-type data storage solution. For example, ApsaraDB for RDS and OSS are used together to implement an online forum. Resources such as the images of registered users and posts on the forum can be stored in OSS to reduce storage needs on ApsaraDB for RDS.

3.5.2. Read/write splitting

This feature allows you to split read requests and write requests across different instances to expand the processing capability of the system.

ApsaraDB RDS for MySQL allows you to directly attach read-only instances to ApsaraDB for RDS to reduce read pressure on the primary instance. The primary instance and read-only instances of ApsaraDB RDS for MySQL each have their own connection endpoints. The system also offers an extra read/write splitting endpoint after read/write splitting is enabled. This endpoint associates the primary instance with all of its read-only instances for automatic read/write splitting, allowing applications to send all read and write requests to a single endpoint. Write requests are automatically routed to the primary instance, and read requests are routed to each read-only instance based on their weights. You can scale out the processing capability of the system by adding more read-only instances. There is no need to modify applications, as shown in Read/write splitting.
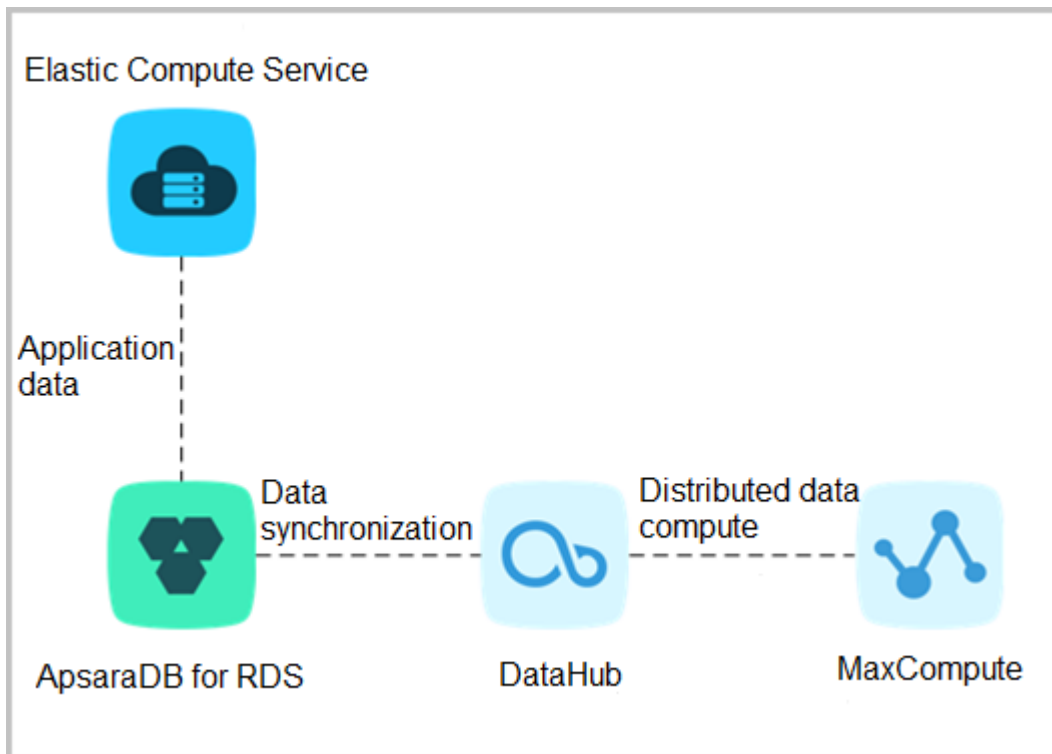
Read/write splitting
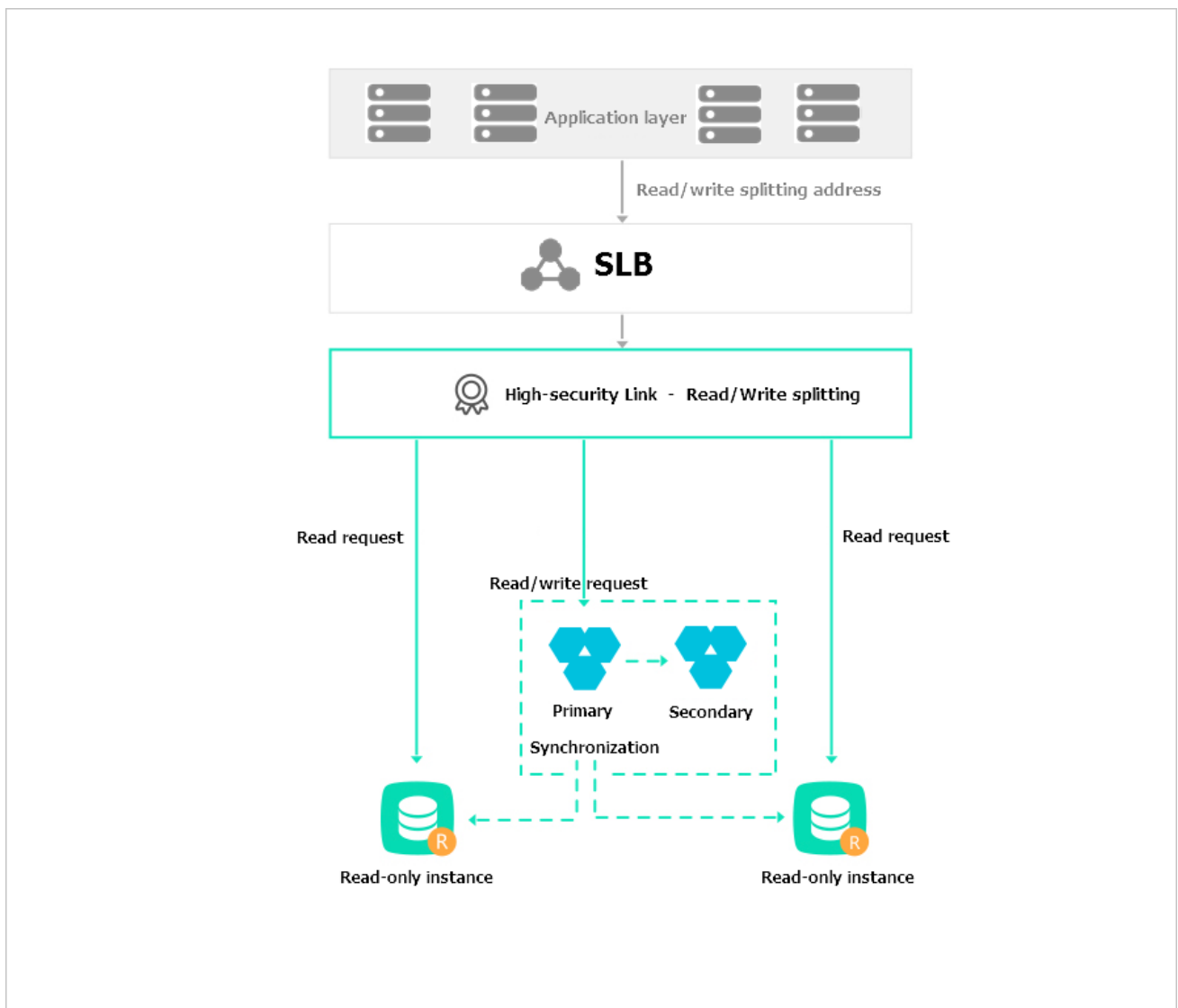
3.5.3. Big data analysis

You can import data from RDS to MaxCompute to enable large-scale data computing.

MaxCompute is used to store and compute batches of structured data. It provides various data

warehouse solutions as well as big data analysis and modeling services, as shown in Big data analysis

diagram.

Big data analysis diagram

3.6. Limits

Before you use ApsaraDB RDS for MySQL, you must understand its limits and take precautions.

To guarantee instance stability and security, ApsaraDB RDS for MySQL has some service limits, as

listed in Limits on ApsaraDB RDS for MySQL.

Limits on ApsaraDB RDS for MySQL

Operation	Description
Database parameter modification	Database parameters can only be modified through the RDS console or API operations. Due to security and stability considerations, only specific parameters can be modified.

Root permissions of databases	The root and SA permissions are not provided.
Database backup	• Logical backup can be performed through the command line interface (CLI) or graphical user interface (GUI). • Physical backup can only be performed through the RDS console or API operations.
Database restoration	• Logical restoration can be performed through the CLI or GUI. • Physical restoration can only be performed through the RDS console or API operations.
Data import	• Logical import can be performed through the CLI or GUI. • Data can be imported through the MySQL CLI or DTS.
ApsaraDB RDS for MySQL storage engine	• Only InnoDB and TokuDB are supported. Due to the inherent shortcomings of the MyISAM engine, some data may be lost. Only some existing instances use the MyISAM engine. MyISAM engine tables in newly created instances will be automatically converted to InnoDB engine tables. • For safety performance and security considerations, we recommend that you use the InnoDB storage engine.

	• The Memory engine is not supported. Newly created Memory tables will be automatically converted into InnoDB tables.
Database replication	ApsaraDB RDS for MySQL provides dual-node clusters based on a primary/secondary replication architecture. The secondary instances in this replication architecture are hidden and cannot be accessed directly.
RDS instance restart	Instances must be restarted through the RDS console or API operations.
Account and database management	ApsaraDB RDS for MySQL uses the RDS console to manage accounts and databases. ApsaraDB RDS for MySQL also allows you to create a privileged account to manage users, passwords, and databases.
Standard account	• Custom authorization is not supported. • The account management and database management interfaces are provided in the RDS console. • Instances that support standard accounts also support privileged accounts.
Privileged account	• Custom authorization is supported.

	• The RDS console does not provide interfaces to manage accounts or databases. Relevant operations can only be performed through code or DMS. • The privileged account cannot be reverted back to a standard account.

3.7. Terms

Term	Description
region	The geographical location where the server of your RDS instance resides. You must specify a region when you create an RDS instance. The region of an instance cannot be changed after instance creation. RDS must be used together with ECS and only supports internal access. Because of this, RDS instances must be located in the same region as their corresponding ECS instances.
zone	The physical area with an independent power supply and network in a region. Zones in a region can communicate through the internal network. Network latency for resources within the same zone is lower than for those across zones. Faults are isolated between zones. Single zone refers to the case where the three nodes in the RDS instance replica set are all located in the

	same zone. Network latency is reduced if an ECS instance and its corresponding RDS instance are both deployed in the same zone.
instance	The most basic unit of RDS. An instance is the operating environment of ApsaraDB for RDS and works as an independent process on a host. You can create, modify, or delete an RDS instance in the RDS console. Instances are mutually independent and their resources are isolated. They do not compete for resources such as CPU, memory, or I/O. Each instance has its own features, such as database type and version. RDS controls instance behavior by using corresponding parameters.
memory	The maximum amount of memory that can be used by an ApsaraDB for RDS instance.
disk capacity	The amount of disk space selected when creating an ApsaraDB for RDS instance. Instance data that occupies disk space includes aggregated data as well as data required for normal instance operations such as system databases, database rollback logs, redo logs, and indexing. Ensure that the disk capacity is sufficient for the RDS instance to store data. Otherwise, the RDS may be locked. If the instance is locked due to insufficient disk capacity, you can unlock the instance by expanding the disk capacity.

IOPS	The maximum number of read/write operations performed per second on block devices at a granularity of 4 KB.
CPU core	The maximum computing capability of the instance. A single Intel Xeon series CPU core has at least 2.3 GHz of computational power with hyper-threading capabilities.
number of connections	The number of TCP connections between a client and an RDS instance. If the client uses a connection pool, the connection between the client and RDS instance is a persistent connection. Otherwise, it is a short-lived connection.

4.Data Transmission Service (DTS)

4.1. What is DTS?

Data Transmission Service (DTS) is a data service that is provided by Alibaba Cloud. DTS supports

data transmission between various types of data sources, such as relational databases.

Features

DTS has the following advantages over traditional data migration and synchronization tools: high

compatibility, high performance, security, reliability, and ease of use. DTS allows you to simplify data

transmission and focus on business development.

Feature	Description
Data migration	You can use DTS to migrate data between homogeneous and heterogeneous data sources. This feature applies to the following scenarios: data migration to Alibaba Cloud, data migration between instances within Alibaba Cloud, and database splitting and scale-out.
Data synchronization	You can use DTS to synchronize data between data sources. This feature applies to the following scenarios: disaster recovery, data backup, load balancing, cloud BI systems, and real-time data warehousing.
Change tracking	You can use DTS to track data changes from databases in real time. This feature applies to the following scenarios: cache updates, business

	decoupling, asynchronous data processing, synchronization of heterogeneous data, and synchronization of extract, transform, and load (ETL) operations.

4.2. Benefits

Data Transmission Service (DTS) allows you to transfer data between various data sources, such as

relational databases and online analytical processing (OLAP) databases. DTS provides the following

data transmission methods: data migration, data synchronization, and change tracking. Compared with

other data migration and synchronization tools, DTS provides transmission channels with higher

compatibility, performance, security, and reliability. DTS also provides a variety of features to help you

create and manage transmission channels.

High compatibility

DTS allows you to migrate or synchronize data between homogeneous and heterogeneous data

sources. For migration between heterogeneous data sources, DTS supports schema conversion.

DTS provides the following data transmission methods: data migration, data synchronization, and

change tracking. In change tracking and data synchronization, data is transferred in real time.

DTS minimizes the impact of data migration on applications to ensure service continuity. The

application downtime during data migration is minimized to several seconds.

High performance

DTS uses high-end servers to ensure the performance of each data synchronization or migration

channel.

DTS uses a variety of optimization measures for data migration.

Compared with traditional data synchronization, the data synchronization feature of DTS refines the

granularity of concurrency to the transaction level. The feature allows you to synchronize incremental

data in one table by using multiple concurrent channels. This improves synchronization performance.

Security and reliability

DTS is implemented based on clusters. If a node in a cluster is unavailable or faulty, the control center

switches all tasks on this node to another node in the cluster.

Secure transmission protocols and tokens are used for authentication across DTS modules to ensure

reliable data transmission.

Ease of use

The DTS console provides a codeless wizard for you to create and manage channels.

To facilitate channel management, the DTS console shows information about transmission channels,

such as transmission status, progress, and performance.

DTS supports resumable transmission, and monitors channel status on a regular basis. If DTS detects

a network failure or system error, DTS automatically fixes the failure or error and restarts the channel. If

the failure or error persists, you must manually repair and restart the channel in the DTS console.

4.3. Environment requirements

You must use DTS on hosts of the following models:

- PF51. *

- PV52P2M1. *

- DTS_E. *

- PF61. *

- PF61P1. *

- PV62P2M1. *

- PV52P1. *

- Q5F53M1. *

- PF52M2. *

- Q41. *

- Q5N1.22

- Q5N1.2B

- Q46.22

- Q46.2B

- W41.22

- W41.2B

- W1.22

- W1.2B

- W1.2C

- D13.12

You must use the following operating system:

AliOS7U2-x86-64

Notice

- Do not use DTS on hosts that are excluded from the preceding models.

- The */apsara* directory used by DTS resides on only one hard disk. Make sure that the

  available space in the directory is larger than 2 TB.

  If the available space in the */apsara* directory is less than 2 TB, tasks cannot run as

  expected and errors will occur. If a task fails, the task recovery and data pulling are affected.
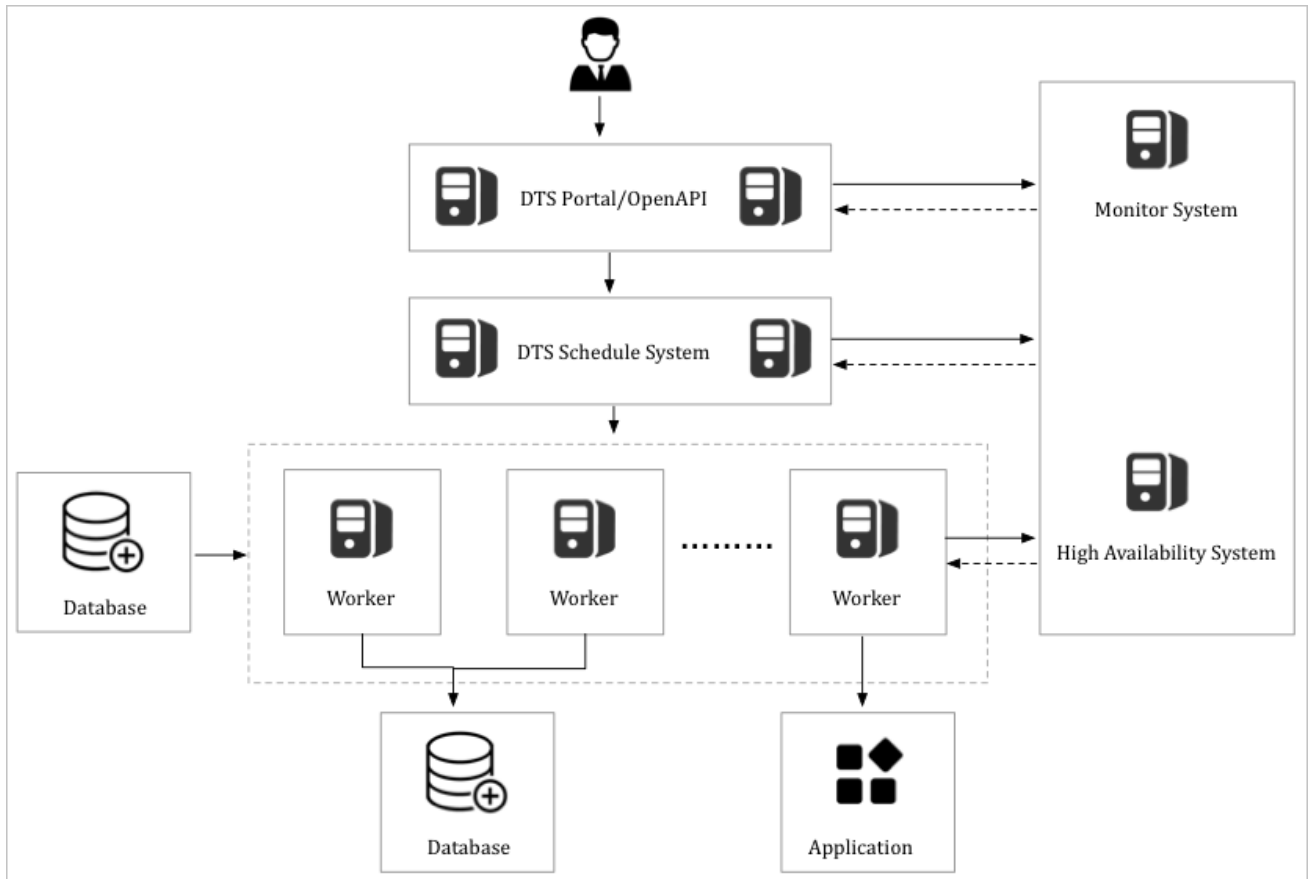
4.4. Architecture

This topic introduces the system architecture of Data Transmission Service (DTS) and the design

concepts of DTS features.

System architecture

The following figure shows the system architecture of Data Transmission Service (DTS).

System architecture

- High availability

  Each module in DTS has a primary node and a secondary node to ensure high availability. The

  disaster recovery module runs a health check on each node in real time. If a node failure is

  detected, the module switches the channel to a healthy node within only a few seconds.

- Connection reliability

  To ensure the connection reliability of change tracking and data synchronization channels, the

  disaster recovery module checks for configuration changes, such as changes of a data source

  address. If a data source address is changed, the module allocates a new connection method to

  ensure the stability of the channel.

Design concept of data migration

The following figure shows the design concept of data migration.

Design concept of data migration
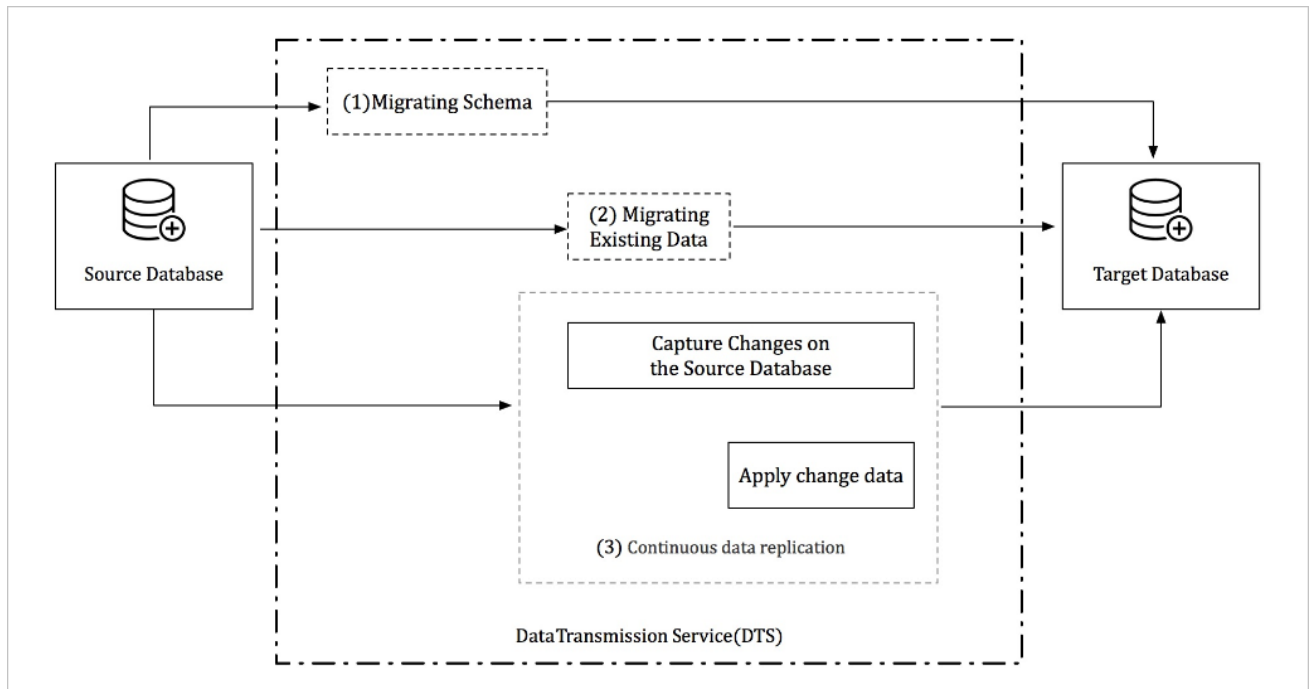
Data migration supports schema migration, full data migration, and incremental data migration. The

following processes ensure service continuity during data migration:

1. Schema migration

2. Full data migration

3. Incremental data migration

To migrate data between heterogeneous databases, DTS reads the source database schema, converts

the schema into the syntax of the destination database, and imports the schema to the destination

database.

A full data migration requires a long period of time. During this process, incremental data is

continuously written to the source database. To ensure data consistency, DTS starts the incremental

data reading module before full data migration. This module retrieves incremental data from the source

database, and parses, encapsulates, and locally stores the data.

After the full data migration is complete, DTS starts the incremental data loading module. This module

retrieves incremental data from the incremental data reading module. After reverse parsing, filtering,

and encapsulation, incremental data is migrated to the destination database in real time.

Design concept of change tracking

The following figure shows the design concept of change tracking.

Design concept of change tracking
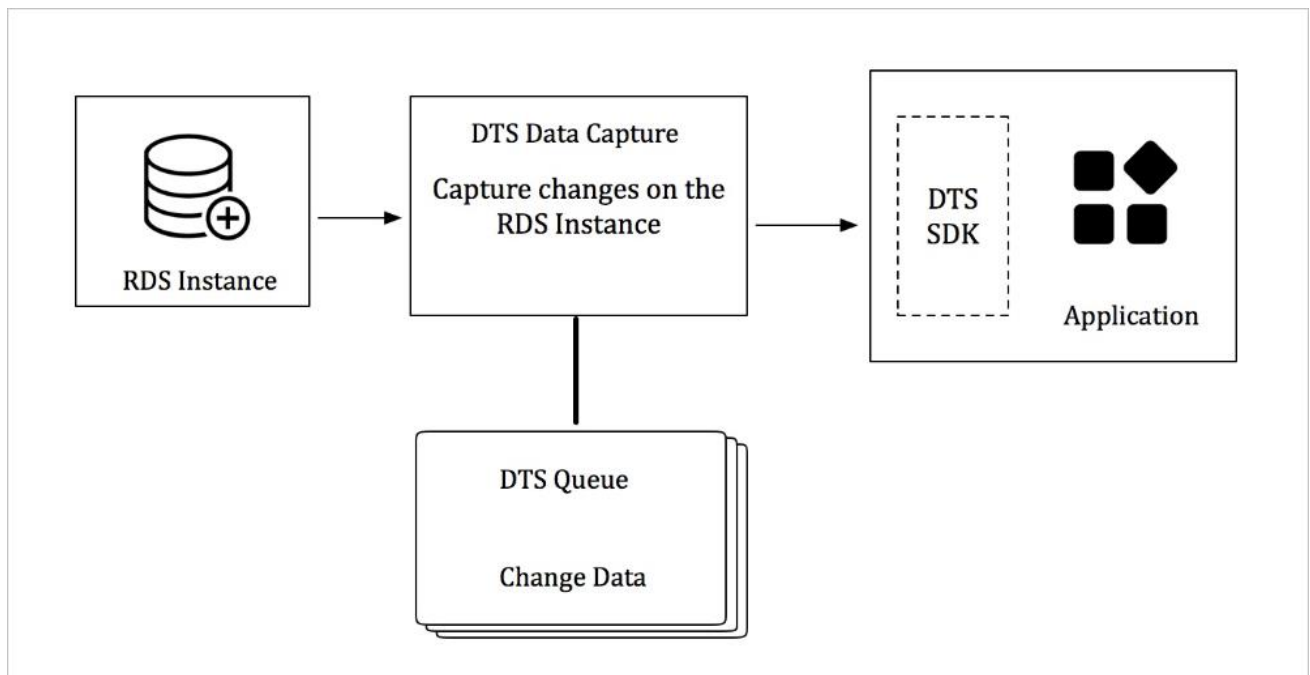
The change tracking feature allows you to obtain incremental data from an RDS instance in real time.

You can subscribe to incremental data on the change tracking server by using DTS SDKs. You can

also customize data consumption rules based on your business requirements.

The incremental data reading module on the server side of DTS retrieves raw data from the source instance. After parsing, filtering, and syntax conversion, incremental data is locally stored.

The incremental data reading module connects to the source instance by using a database protocol and retrieves incremental data from the source instance in real time. If the source instance is an ApsaraDB RDS for MySQL instance, the incremental data reading module connects to the source instance by using the binary log dump protocol.

DTS ensures high availability of the incremental data reading module and consumption SDK processes.

If an error is detected in the incremental data reading module, the disaster recovery module restarts the incremental data reading module on a healthy node. This ensures high availability of the incremental data reading module.

DTS ensures high availability of consumption SDK processes on the server. If you start multiple consumption SDK processes for the same change tracking channel, the server pushes incremental data to only one process at a time. If an error occurs on a process, the server pushes data to another healthy consumption process.

Design concept of data synchronization

The following figure shows the design concept of data synchronization.

Design concept of data synchronization
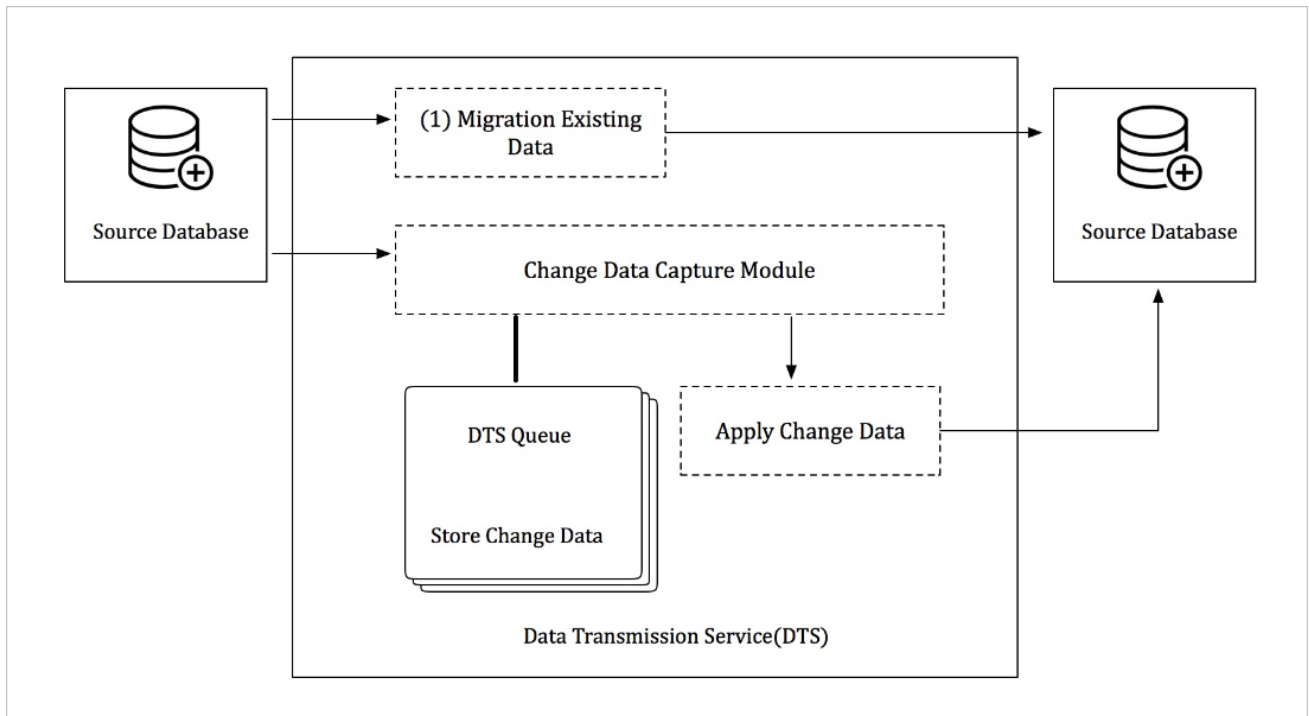
The data synchronization feature can be used to synchronize incremental data between two RDS instances.

A data synchronization channel is established by using the following processes:

- Initial data synchronization: DTS synchronizes historical data from the source instance to the destination instance.

- Incremental data synchronization: After initial data synchronization, DTS synchronizes incremental data from the source instance to the destination instance.

DTS synchronizes incremental data by using the following modules:

- Incremental data reading module

  The incremental data reading module retrieves raw data from the source instance. After parsing, filtering, and syntax conversion, the data is locally stored. The incremental data reading module connects to the source instance by using a database protocol and obtains incremental data from

the source instance. If the source instance is an ApsaraDB RDS for MySQL instance, the

incremental data reading module connects to the source instance by using the binary log dump

protocol.

- Incremental data loading module

  The incremental data loading module retrieves incremental data from the incremental data reading

  module and filters data based on the required objects. Then, the incremental data loading module

  synchronizes data to the destination instance without compromising transactional sequence and

  consistency.

DTS ensures high availability of the incremental data reading module and incremental data loading

module. If a channel failure is detected, the disaster recovery module switches the channel to a healthy

node. This ensures high availability of the synchronization channel.

4.5. Features

4.5.1. Data migration

You can use Data Transmission Service (DTS) to migrate data between various types of data sources.

Typical scenarios include data migration to the cloud, data migration between instances within Apsara

Stack, and database splitting and scale-out. Data migration supports the following extract, transform,

and load (ETL) features: object name mapping and data filtering.

Database and migration types

Source database	Destination database	Migration type

User-created MySQL database Version 5.5, 5.6, 5.7, or 8.0	User-created MySQL database Version 5.5, 5.6, 5.7, or 8.0	• Schema migration • Full data migration • Incremental data migration
	User-created Kafka database Version 0.10.1.0 to 1.0.2	
User-created PostgreSQL database Version 9.4, 9.5, 9.6, or 10.x	User-created PostgreSQL database Version 9.4, 9.5, 9.6, or 10.x	
User-created Oracle database Version 9i, 10g, 11g, 12c, 18c, or 19c	AnalyticDB for PostgreSQL Version 4.3 or 6.0	
User-created MongoDB database	User-created MongoDB database	

| Version 3.0, 3.2, 3.4, 3.6, or 4.0 | Version 3.0, 3.2, 3.4, 3.6, or 4.0 | • Full data migration |
| User-created Redis database Version 2.8, 3.0, 3.2, or 4.0 | User-created Redis database Version 2.8, 3.0, 3.2, or 4.0 | • Incremental data migration **Note** MongoDB and Redis are NoSQL databases that do not require schema migration. |

Online migration

DTS uses online migration. You must configure the source instance, destination instance, and objects to be migrated. DTS automatically completes the entire data migration process. You can select all of the supported migration types to minimize the impact of online data migration on your services. However, you must ensure that DTS servers can connect to both the source and destination instances.

Data migration types

DTS supports schema migration, full data migration, and incremental data migration.

- Schema migration: DTS migrates schemas from the source instance to the destination instance.

- Full data migration: migrates historical data from the source instance to the destination instance.

- Incremental data migration: DTS synchronizes incremental data that is generated during data

  migration from the source instance to the destination instance. You can select schema migration,

  full data migration, and incremental migration to migrate data with minimal downtime.

ETL features

Data migration supports the following ETL features:

- Object name mapping: You can change the names of the columns, tables, and databases that are

  migrated to the destination database.

- Data filtering: You can use SQL conditions to filter the required data in a specific table. For

  example, you can specify a time range to migrate only the latest data.

Alerts

If an error occurs during data migration, DTS immediately sends an SMS alert to the task owner. This

allows the owner to handle the error at the earliest opportunity.

Migration task

A migration task is a basic unit of data migration. To migrate data, you must create a migration task in

the DTS console. To create a migration task, you must configure the required information such as the

source and destination instances, migration types, and objects to be migrated. You can create,

manage, stop, and delete migration tasks in the DTS console.

The following table describes the statuses of a migration task.

Task statuses

Status	Description	Available operation
Not Started	The migration task has been configured but no precheck is performed.	- Run a precheck - Delete the migration task
Prechecking	A precheck is being performed but the migration task is not started.	Delete the migration task
Passed	The migration task has passed the precheck but has not been started.	- Start the migration task - Delete the migration task

Migrating	The task is migrating data.	• Pause the migration task • Stop the migration task • Delete the migration task
Migration Failed	An error occurred during data migration. You can identify the point of failure based on the progress of the migration task.	Delete the migration task
Paused	The migration task is paused.	• Start the migration task • Delete the

| | | migration task |
| | | |

| Completed | The migration task is completed, or you have stopped data migration by clicking **End**. | Delete the migration task |

4.5.2. Data synchronization

You can use Data Transmission Service (DTS) to synchronize data between two data sources. This

feature applies to various scenarios, such as data backup, disaster recovery, active geo-redundancy,

cross-border data synchronization, load balancing, cloud BI systems, and real-time data warehousing.

Supported databases

Source database	Destination database	Initial synchronization type	Synchronization topology
• User-created MySQL database Version 5.5, 5.6, 5.7, or 8.0	User-created MySQL database Version 5.5, 5.6, 5.7, or 8.0	Initial schema synchronization Initial full data synchronization	One-way synchronization Two-way synchronization

RDS MySQL Version 5.6 or 5.7	RDS MySQL Version 5.6 or 5.7	Initial schema synchronization Initial full data synchronization	One-way synchronization Two-way synchronization
PolarDB-X (formerly known as DRDS)	PolarDB-X	Initial full data synchronization	One-way synchronization

Objects to be synchronized

- You can select columns, tables, or databases as the objects to be synchronized. You can specify one or more tables that you want to synchronize.

- DTS allows you to synchronize data between tables that have different names, or between databases that have different names. You can use the object name mapping feature to specify the names of destination columns, tables, and databases.

- You can specify one or more columns that you want to synchronize.

Synchronization tasks

A synchronization task is a basic unit of data synchronization. To synchronize data between two instances, you must create a synchronization task in the DTS console.

The following table describes the statuses of a synchronization task.

Task statuses

Task status	Description	Available operation
Prechecking	A precheck is being performed before the synchronization task is started.	- View the configurations of the synchronization task - Delete the synchronization task - Replicate the configurations of the synchronization task - Configure monitoring and alerts
Precheck Failed	The synchronization task has failed to pass the precheck.	- Run a precheck - View the configurations of the synchronization task - Reselect the objects to be synchronized - Modify the synchronization speed - Delete the synchronization task

		- Replicate the configurations of the synchronization task - Configure monitoring and alerts
Not Started	The synchronization task has passed the precheck but has not been started.	- Run a precheck - Start the synchronization task - Reselect the objects to be synchronized - Modify the synchronization speed - Delete the synchronization task - Replicate the configurations of the synchronization task - Configure monitoring and alerts

Performing Initial Synchronization	Initial synchronization is being performed.	• View the configurations of the synchronization task • Delete the synchronization task • Replicate the configurations of the synchronization task • Configure monitoring and alerts
Initial Synchronization Failed	The task has failed during initial synchronization.	• View the configurations of the synchronization task • Reselect the objects to be synchronized • Modify the synchronization speed • Delete the synchronization task

		- Replicate the configurations of the synchronization task - Configure monitoring and alerts
Synchronizing	The task is synchronizing data.	- View the configurations of the synchronization task - Reselect the objects to be synchronized - Modify the synchronization speed - Pause the synchronization task - Delete the synchronization task - Replicate the configurations of the synchronization task

		• Configure monitoring and alerts
Synchronization Failed	An error occurred during synchronization.	• View the configurations of the synchronization task • Reselect the objects to be synchronized • Modify the synchronization speed • Start the synchronization task • Delete the synchronization task • Replicate the configurations of the synchronization task • Configure monitoring and alerts
Paused	The synchronization task is paused.	• View the configurations of the synchronization task

		• Reselect the objects to be synchronized • Modify the synchronization speed • Start the synchronization task • Delete the synchronization task • Replicate the configurations of the synchronization task • Configure monitoring and alerts

Advanced features

You can use the following advanced features to facilitate data synchronization:

- Add or remove the objects to be synchronized

  You can add or remove the required objects when a task is synchronizing data.

- View and analyze the synchronization performance

DTS provides trend charts that allow you to view and analyze the performance of your

synchronization tasks. The synchronization performance is measured based on bandwidth,

synchronization speed (RPS), and synchronization delay.

- Monitor synchronization tasks

DTS allows you to monitor the status of synchronization tasks. If the threshold for synchronization

delay is reached, you will receive an alert. You can set the alert threshold based on the sensitivity

of your businesses to synchronization delays.

4.5.3. Change tracking

You can use Data Transmission Service (DTS) to track data changes from user-created MySQL

databases in real time. This feature applies to the following scenarios: cache updates, business

decoupling, synchronization of heterogeneous data, and synchronization of extract, transform, and load

(ETL) operations.

Supported databases

- User-created MySQL database

- User-created Oracle database

Objects for change tracking

The objects for change tracking include tables and databases. You can specify one or more tables from

which you want to track data changes.

In change tracking, data changes include data manipulation language (DML) operations and data definition language (DDL) operations. When you configure a change tracking task, you can select the operation type.

Change tracking tasks

A change tracking task is the basic unit of change tracking and data consumption. To track data changes from a MySQL database, you must create a change tracking task for the MySQL database in the DTS console. The change tracking task pulls incremental data from the MySQL database in real time and locally stores the incremental data. You can use the DTS SDK to consume the incremental data from the change tracking task. You can also create, manage, or delete change tracking tasks in the DTS console.

A change tracking task can be consumed by only one downstream SDK client. To track data changes from a MySQL database by using multiple downstream SDK clients, you must create an equivalent number of change tracking tasks.

The Task statuses table describes the statuses of a change tracking task.

Task statuses

Task status	Description	Available operation
Prechecking	The change tracking task has been configured and a precheck is being performed.	Delete the change tracking task

Not Started	The change tracking task has passed the precheck but has not been started.	- Start the change tracking task - Delete the change tracking task
Performing Initial Change Tracking	The initial change tracking is in progress. This process takes about 1 minute.	Delete the change tracking task
Normal	Incremental data is being pulled from the MySQL database.	- View the demo code - View the tracked data - Delete the change tracking task
Error	An error occurs when the change tracking task is pulling incremental data from the MySQL database.	- View the demo code - Delete the change tracking task

Advanced features

You can use the following advanced features that are provided for change tracking:

- Add or remove the objects for change tracking

  You can add or remove the required objects when a change tracking task is running.

- View the tracked data

  You can view the data that is tracked by the change tracking task in the DTS console.

- Modify consumption checkpoints

  You can modify consumption checkpoints.

- Monitor change tracking tasks

  DTS allows you to monitor the status of change tracking tasks. If the threshold for consumption delay is reached, you will receive an alert. You can set the alert threshold based on the sensitivity of your businesses to consumption delays.

4.6. Scenarios

Data Transmission Service (DTS) supports data migration, change tracking, and data synchronization in various scenarios.

Database migration with minimized downtime

To ensure data consistency, traditional migration requires that you stop writing data to the source database during data migration. Depending on the data volume and network conditions, the migration may take several hours or even days, which has a great impact on your businesses.

DTS provides migration with minimized downtime. Services are always available except when they are switched from the source instance to the destination instance. The service downtime is minimized to minutes. The following figure shows the architecture of data migration.
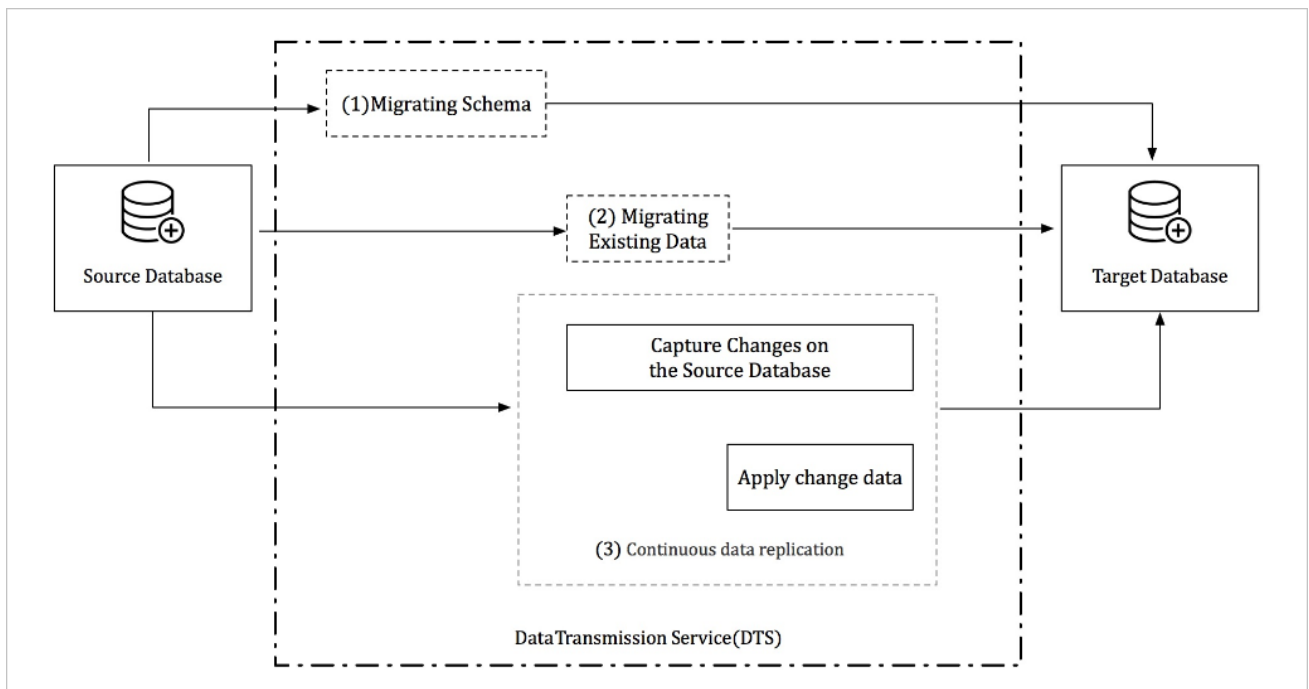
The data migration process includes schema migration, full data migration, and incremental data migration. During incremental data migration, the data in the source instance is synchronized to the destination instance in real time. You can verify businesses in the destination database. After the verification succeeds, you can migrate businesses to the destination database.

Real-time data analysis

Data analysis is essential in improving enterprise insights and user experience. With real-time data analysis, enterprises can adjust marketing strategies to adapt to changing markets and higher demands for better user experience.

With the change tracking feature provided by DTS, you can acquire real-time incremental data without affecting online businesses. You can use the DTS SDK to synchronize the subscribed incremental data to the analysis system for real-time analysis.
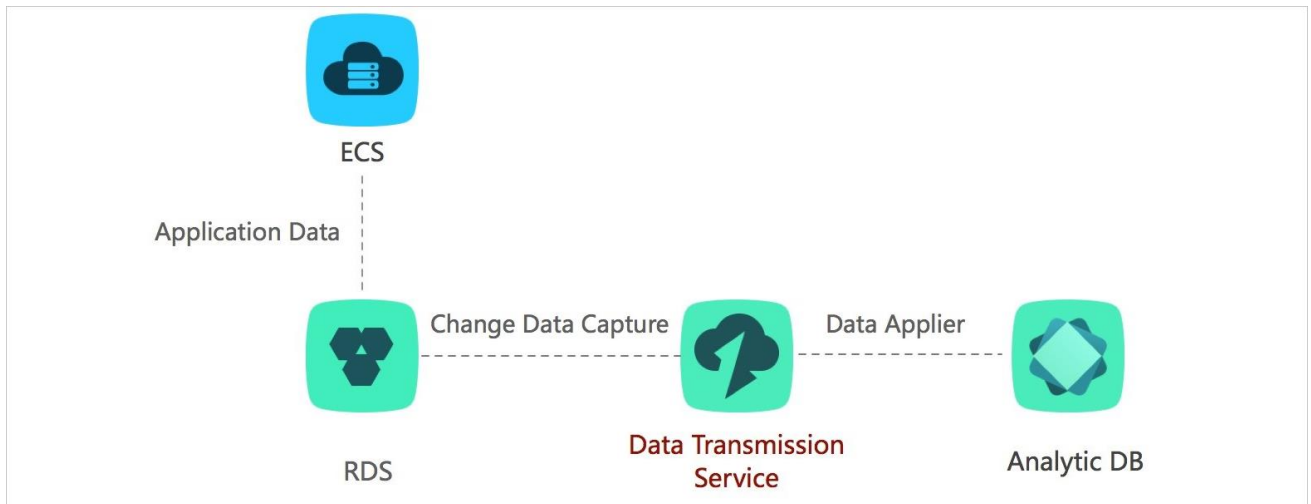
Lightweight cache update policies

To accelerate access speed and improve concurrent read performance, a cache layer is used in the business architecture to receive all read requests. The memory read mechanism of the cache layer can help to improve read performance. The data in the cache memory is not persistent. If the cache memory fails, the data in the cache memory will be lost.

With the change tracking feature provided by DTS, you can subscribe to the incremental data in databases and update the cached data to implement lightweight cache update policies.
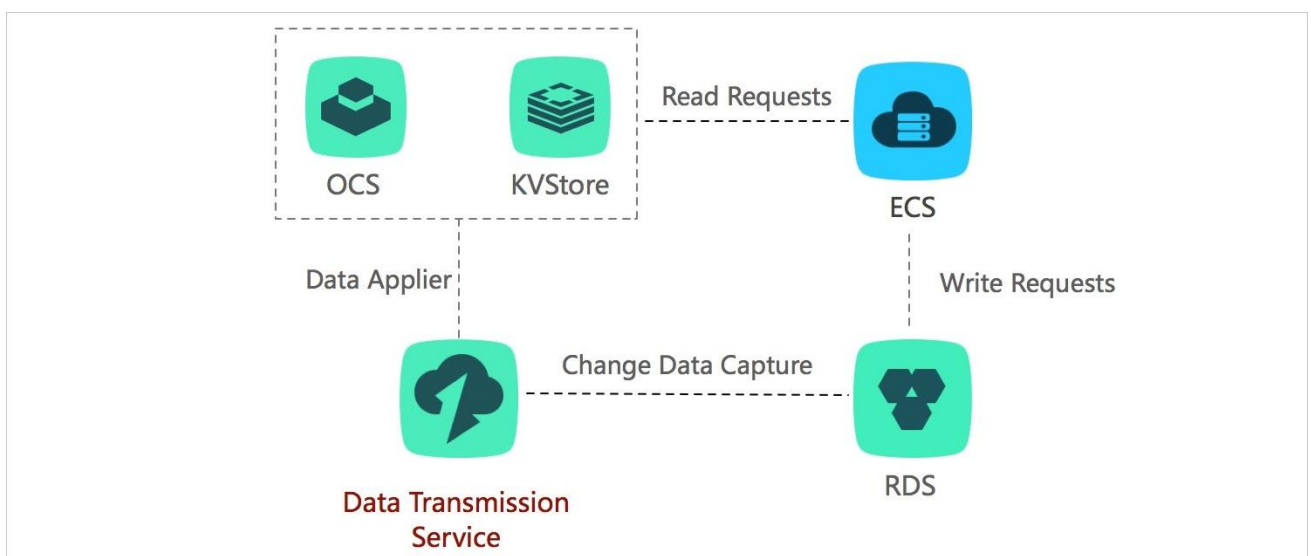
Benefits

- Quick update with low latency

  The business returns data after the database update is complete. For this reason, you do not need to consider the cache invalidation process, and the entire update path is short with low latency.

- Simple and reliable applications

  The complex doublewrite logic is not required for the applications. You only need to start the asynchronous thread to monitor the incremental data and update the cached data.

- Application updates without extra performance consumption

  DTS retrieves incremental data by parsing incremental logs in the database, which does not affect the performance of businesses and databases.

Business decoupling

The e-commerce industry involves many different types of business logic such as ordering, inventory, and logistics. If all of these types of business logic are included in the ordering process, the order result can be returned only after all the changes are complete. However, this may cause the following issues:

- The ordering process consumes a long period of time and results in poor user experience.

- The business system is unstable and downstream faults will affect service availability.

With the change tracking feature provided by DTS, you can optimize your business system and receive notifications in real time. You can decouple different types of business logic and asynchronously

process data. This makes the core business logic simpler and more reliable. The following figure shows

the architecture of business decoupling.
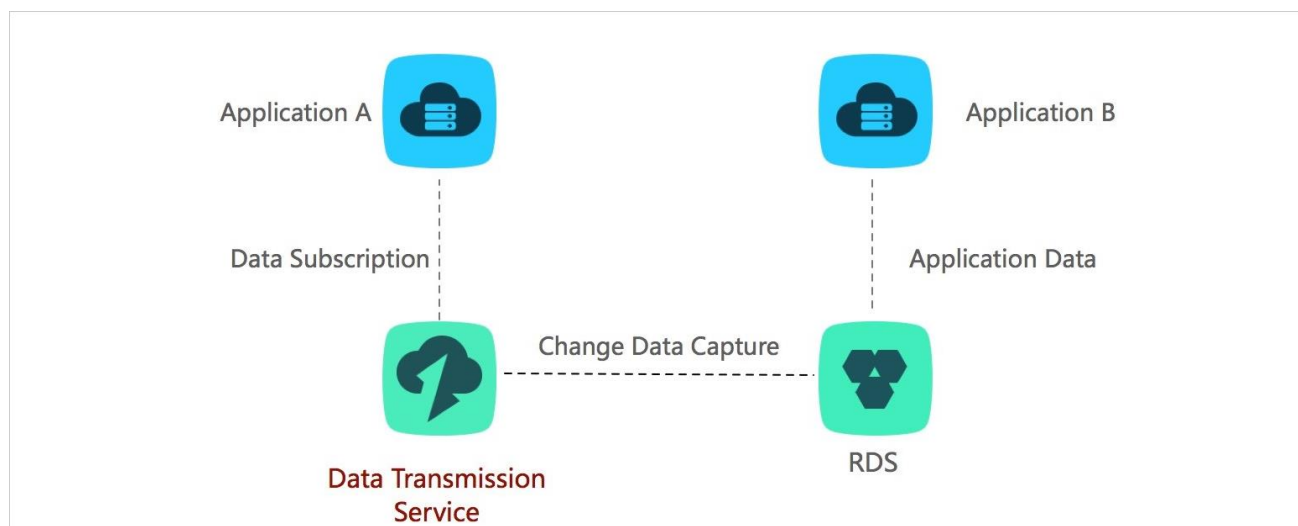
In this scenario, the ordering system returns the result after the buyer places an order. The underlying

layer obtains the data changes that are generated in the ordering system in real time by using the

change tracking feature. You can subscribe to these data changes by using the DTS SDK, which

triggers different types of downstream business logic such as inventory and logistics. This ensures that

the entire business system is simple and reliable.

This scenario has been applied to a wide range of businesses in Alibaba Group. Tens of thousands of

downstream businesses in the Taobao ordering system are using the change tracking feature to

retrieve real-time data updates and trigger business logic every day.

4.7. Terms

This topic describes the terms that are used in the DTS documentation.

Term	Description

precheck	The system performs a precheck before starting a data migration task, data synchronization task, or change tracking task. For example, the following items are checked: the connectivity between DTS servers and the source and destination databases, database account permissions, whether binary logging is enabled, and database version numbers. **Note** If a task fails to pass the precheck, you can click the ⓘ icon next to each failed item to view the related details. Then, you can troubleshoot the issues based on the causes and run a precheck again.
schema migration	DTS migrates the schemas of the required objects from the source database to the destination database. Tables, views, triggers, and stored procedures can be migrated. For schema migration between heterogeneous databases, DTS converts the schema syntax based on the syntax of the source and destination databases. For example, it converts the NUMBER data type in Oracle databases into the DECIMAL data type in MySQL databases.
full data migration	DTS migrates historical data of the required objects from the source database to the destination database. If you select only schema migration and full data migration, incremental data that was generated in the source database is not migrated to the

	destination database. To ensure data consistency, do not write data to the source database during full data migration. Note To ensure service continuity, you must select schema migration, full data migration, and incremental data migration when you configure a data migration task.
incremental data migration	DTS retrieves static snapshots from the source database and migrates the snapshot data to the destination database. Then, DTS synchronizes incremental data that was generated in the source database to the destination database. Note During incremental data migration, data between the source and destination databases is synchronized in real time. The migration task does not automatically stop. You must manually stop the migration task.
initial synchronization	Before DTS synchronizes incremental data, DTS synchronizes the schemas and historical data of the required objects to the destination database. Initial synchronization includes initial schema synchronization and initial full data synchronization. During initial schema synchronization, DTS synchronizes the schemas of the required objects from the source database to the destination database. During initial full data synchronization, DTS

	synchronizes historical data of the required objects from the source database to the destination database.
synchronization performance	The synchronization performance is measured by the number of transactions that are synchronized to the destination database per second.
synchronization delay	The synchronization delay is the difference between the timestamp of the latest data that is synchronized to the destination database and the current timestamp of the source database. The synchronization delay indicates the time difference between the source and destination databases for the latest data. If the synchronization delay is zero, the data in the source database is consistent with the data in the destination database.
data update	Data updates are operations that modify data without modifying the schema, such as the INSERT, DELETE, and UPDATE operations.
schema update	Schema updates are operations that modify the schema syntax, such as the CREATE TABLE, ALTER TABLE, and DROP VIEW operations.
timestamp range	The timestamp range is the range of timestamps for incremental data that is stored in a change tracking channel. By default, the change tracking channel retains the data that was generated in the most recent

|	
	24 hours. DTS clears expired incremental data on a regular basis and updates the timestamp range of the change tracking channel. Note The timestamp of incremental data is generated when the data is updated in the source database and written to the transaction log.
consumption checkpoint	The consumption checkpoint is the timestamp of the latest incremental data that is consumed by a downstream SDK client. When the SDK client consumes a data record, it returns a confirmation message to DTS. DTS updates and saves the consumption checkpoint. If the SDK client restarts due to exceptions, DTS pushes incremental data from the last consumption checkpoint.

5.Cloud Native Distributed Database PolarDB-X

5.1. What is PolarDB-X?

Cloud Native Distributed Database PolarDB-X (PolarDB-X) is a middleware service independently

developed by Alibaba Group for scale-out of single-instance relational databases. It is compatible with

Distributed Relational Database Service (DRDS). Compatible with the MySQL protocol, PolarDB-X

supports most MySQL data manipulation language (DML) and data definition language (DDL) syntax. It

provides the core capabilities of distributed databases, such as database sharding, table sharding,

smooth scale-out, configuration changing, and transparent read/write splitting. PolarDB-X features

lightweight (stateless), flexibility, stability, and efficiency, and provides you with O&M capabilities

throughout the lifecycle of distributed databases.

PolarDB-X is mainly used for operations on large-scale online data. By splitting data in specific

business scenarios, PolarDB-X maximizes the operation efficiency, meeting the requirements of online

businesses on relational databases.

Product structure diagram of PolarDB-X
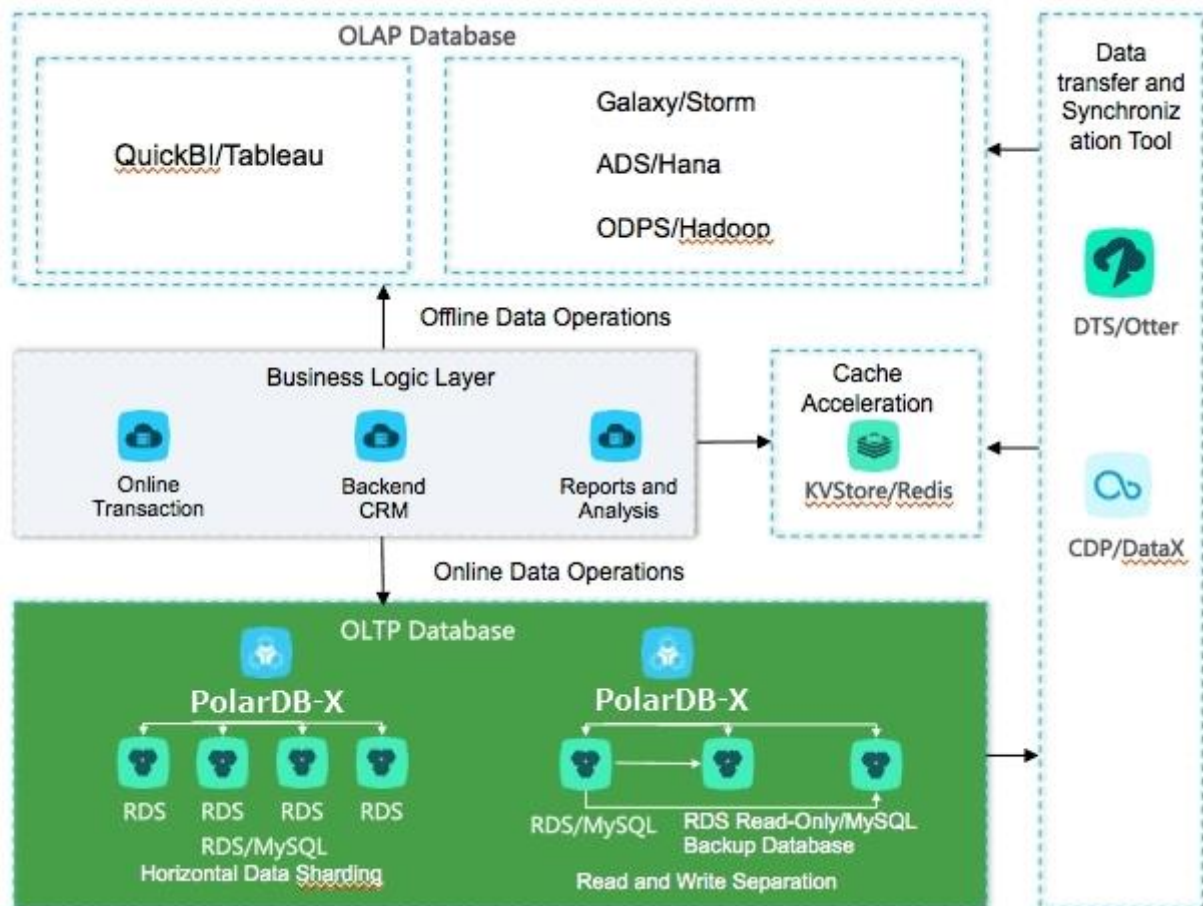
Problems solved

- Capacity bottleneck of single-instance databases: As the data volume and access volume increase, traditional single-instance databases encounter great challenges that cannot be completely solved by hardware upgrades. Distributed solutions use multiple instances to work jointly, effectively resolving the bottlenecks of data storage capacity and access volumes.

- Difficult scale-out of relational databases: Due to the inherent attributes of distributed databases, data can be stored to different shards through smooth data migration, supporting the dynamic scale-out of relational databases.

5.2. Benefits

Distributed architecture

The use of horizontal partitioning and multi-node cluster deployment has helped eliminate single-database resource bottlenecks for Server Load Balancer (SLB), Cloud Native Distributed Database PolarDB-X (PolarDB-X), and ApsaraDB RDS for MySQL. Scalability is also supported, which is the biggest benefit of distributed databases.

Audit logs

The audit logs will store to Log Service (SLS) automatically and you can download logs from SLS. This facilitates long-term storage and management of audit logs.

Auto scaling

PolarDB-X instances and ApsaraDB RDS for MySQL instances can be dynamically added and removed. This allows you to choose service capabilities in a flexible manner.

High performance

PolarDB-X partitions data in specific business scenarios and clusters data based on major operations. Therefore, it speeds up the response to online transactional operations. By using the column store and knowledge grid, PolarDB-X for HiStore significantly speeds up the response to common analytic operations such as large-scale data aggregation and ad hoc queries. PolarDB-X for HiStore also helps reduce costs by achieving a high compression ratio.

High availability of clusters

The multi-node cluster architecture is used. Each component management node in the platform needs to implement a high availability mechanism. Therefore, the failure of a service node in a cluster does not affect the overall operation of the corresponding service instance. Clusters have the adaptive load

balancing capability, which ensures service operation in high-concurrency and high-load scenarios

without requiring you to adjust cluster parameters.

Backup

The data of PolarDB-X is stored in ApsaraDB RDS for MySQL, which supports full or incremental

backup and data recovery from storage. The capability of data cluster backup between data centers is

provided, which allows data backup across data centers. The backup process is visualized.

Disaster recovery

Metadatabases support fast switchover for data recovery. Single-node failures do not affect services.

Security and controllability

PolarDB-X supports an account permission system similar to that of ApsaraDB RDS for MySQL, and

provides useful features, such as the IP address whitelist and default disabling of high-risk SQL

statements. PolarDB-X provides a complete standard API system that can be used by your local

management system. Complete product support and architecture services are also available to you.

Isolation

By using the multi-instance approach, PolarDB-X supports multi-tenant parallel execution in a PolarDB-

X cluster, and tenant tasks are submitted to queues on different instances for execution. PolarDB-X

isolates resources among tenants by using PolarDB-X instances.

Permissions

- PolarDB-X allows you to manage tenants in a centralized manner, dynamically configure and manage tenant resources, isolate resources, view statistics on resource usage, and manage tenants at multiple levels in the console.

- PolarDB-X supports permission management and fine-grained audit of user operations.

- PolarDB-X provides a comprehensive permission authentication and isolation mechanism to ensure your data privacy.

- PolarDB-X allows you to isolate resources by using the multitenancy mode.

Scheduling

PolarDB-X supports multi-tenant scheduling in multiple clusters and resource pools.

5.3. Architecture

Cloud Native Distributed Database PolarDB-X (PolarDB-X) supports two data output methods: overall output by Apsara Stack and separate output by Alibaba middleware. The two output methods differ in features and dependent components of PolarDB-X.

The following table describes the differences between these two methods.

Item	Overall output by Apsara Stack	Separate output by Alibaba middleware
MySQL	ApsaraDB RDS for MySQL	Alibaba Cloud Database Platform as a Service (DBPaaS)

Load balancing	Centralized Server Load Balancer (Centralized SLB)	Client load balancer (VIPServer)
Special storage support	None	High compression-ratio column store (HiStore)

The following figure shows the system architecture of PolarDB-X.

PolarDB-X system architecture
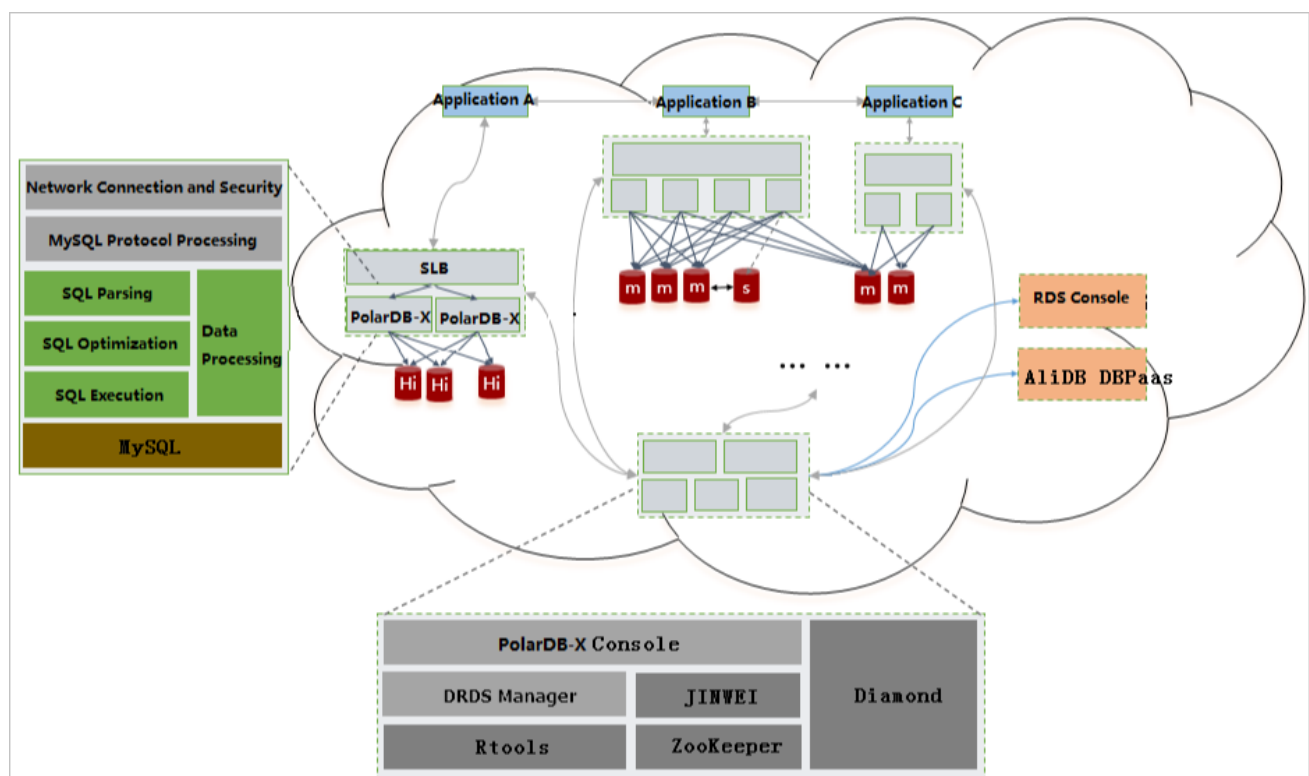
PolarDB-X Server

PolarDB-X Server is the service layer of PolarDB-X. Multiple PolarDB-X Server nodes form a cluster to

provide distributed database services, including read/write splitting, routed SQL execution, result

merging, dynamic database configuration, and globally unique ID (GUID).

Note PolarDB-X instances are stateless. Therefore, ApsaraDB RDS for MySQL instances are used for storage. PolarDB-X implements data encryption by using encryption algorithms such as transparent data encryption (TDE) supported by ApsaraDB RDS for MySQL.

ApsaraDB RDS for MySQL (marked by m and s in the figure)

ApsaraDB RDS for MySQL stores data and performs data operations online. It implements high availability by using primary/secondary replication. It also implements dynamic database failover with the primary/secondary switchover mechanism.

You can implement management, monitoring, and alerting within the instance lifecycle in the ApsaraDB RDS for MySQL console.

HiStore

When PolarDB-X outputs data separately (not overall output by Apsara Stack), it uses HiStore as the physical storage. HiStore is a low-cost and high-performance database developed by Alibaba to support column store. By using the column store, knowledge grid, and multiple cores, HiStore provides higher data aggregation and ad hoc query capabilities, with lower costs than row store (such as MySQL).

You can implement management, monitoring, and alerting within the instance lifecycle in the HiStore console.

DBPaaS

When PolarDB-X outputs data separately (not overall output by Apsara Stack), the ApsaraDB RDS for

MySQL O&M platform DBPaaS implements management, monitoring, alerting, and resource

management in the lifecycle of ApsaraDB RDS for MySQL instances.

Centralized SLB

You do not need to install a client on user instances. SLB is used to distribute your requests. When an

instance fails or a new instance is added, SLB ensures that traffic on the bound instances is distributed

evenly.

VIPServer

You must install a client on user instances, with a weak dependency on the central controller

(interaction is performed only when the load configuration changes). VIPServer is used to distribute

your requests. When an instance fails or a new instance is added, VIPServer ensures that traffic on the

bound instances is distributed evenly.

Diamond

Diamond is a system responsible for PolarDB-X configuration storage and management. It provides the

configuration storage, query, and notification functions. Diamond stores the database source data,

sharding rules, and PolarDB-X switch configuration.

Data Replication System

Data Replication System is responsible for data migration and synchronization of PolarDB-X. The core

capabilities of this system include full data migration and incremental data synchronization. Its derived

features include smooth data import, smooth scale-out, and global secondary index (GSI). Data

Replication System requires the support of ZooKeeper and PolarDB-X Rtools.

PolarDB-X console

PolarDB-X Console is designed for business database administrators (DBAs) to isolate resources as

required and perform operations, such as instance management, database and table management,

read/write splitting configuration, smooth scale-out, monitoring data display, and IP address whitelisting.

DRDS Manager

DRDS Manager is designed for global O&M personnel and DBAs. It provides the PolarDB-X resource

management and system monitoring functions:

- Manages all resources on which ApsaraDB RDS for MySQL instances depend, including virtual

  machines, SLB instances, and domain names.

- Monitors the status of PolarDB-X instances, including queries per second (QPS), active threads,

  connections, node network I/O, and node CPU utilization.

Rtools

Rtools is the O&M support system of PolarDB-X. It allows you to manage database configuration,

read/write weight, connection parameters, database and table topologies, and sharding rules.

5.4. Features

5.4.1. Scalability

Concurrency and storage capacity scalability

The essence of scalability lies in splitting. PolarDB-X distributes data to multiple ApsaraDB RDS for

MySQL instances to obtain the distribution of read/write requests and storage through Horizontal

partitioning. The PolarDB-X layer is stateless and increases nodes to cope with concurrent SQL loads,

which is similar to a business application.

Horizontal partitioning

Data is distributed to multiple ApsaraDB RDS for MySQL instances based on certain calculation and

routing rules. In fact, PolarDB-X has many algorithms to cope with the loads in various scenarios.
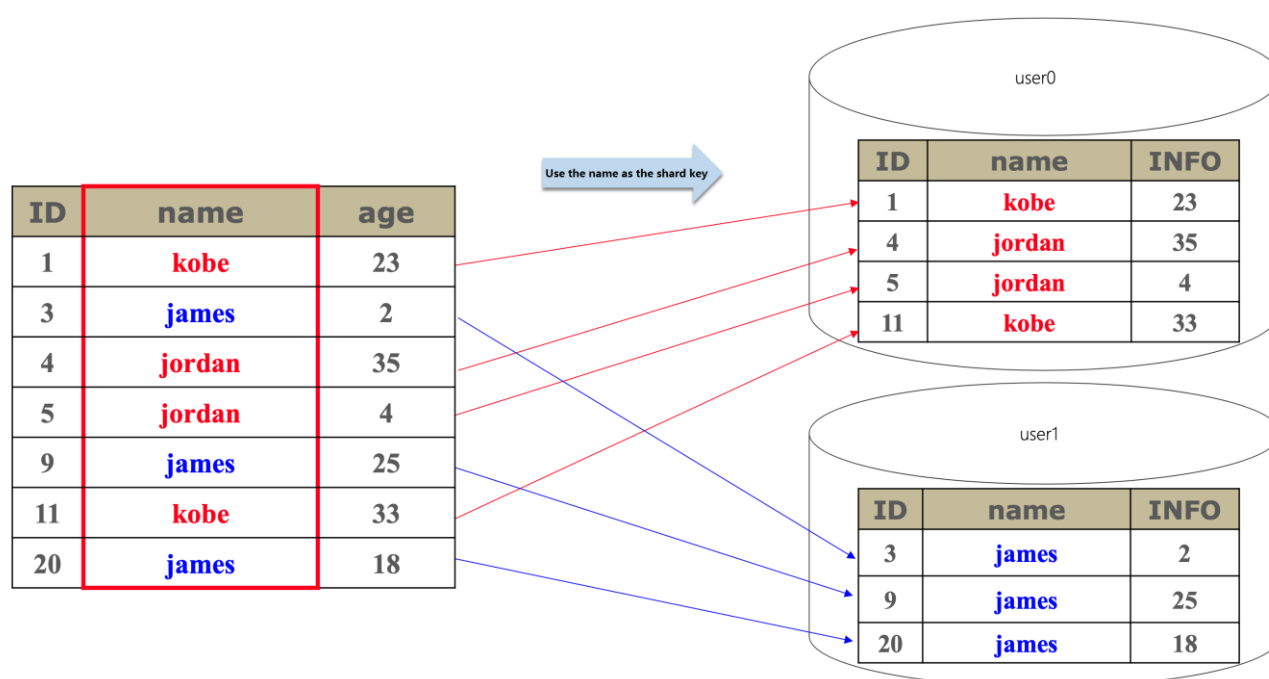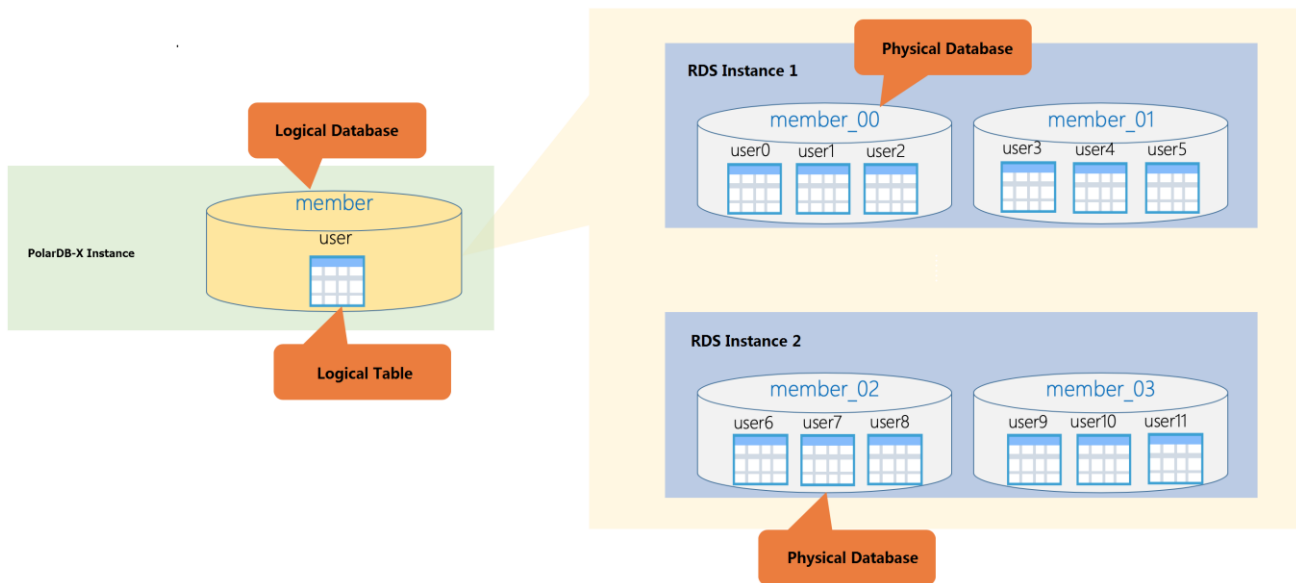
Computing scalability

PolarDB-X often needs to perform complex computing on data far exceeding the capacity of a single

instance. These SQL statements include multi-table join, multi-layer nested subqueries, grouping,

sorting, and aggregation.
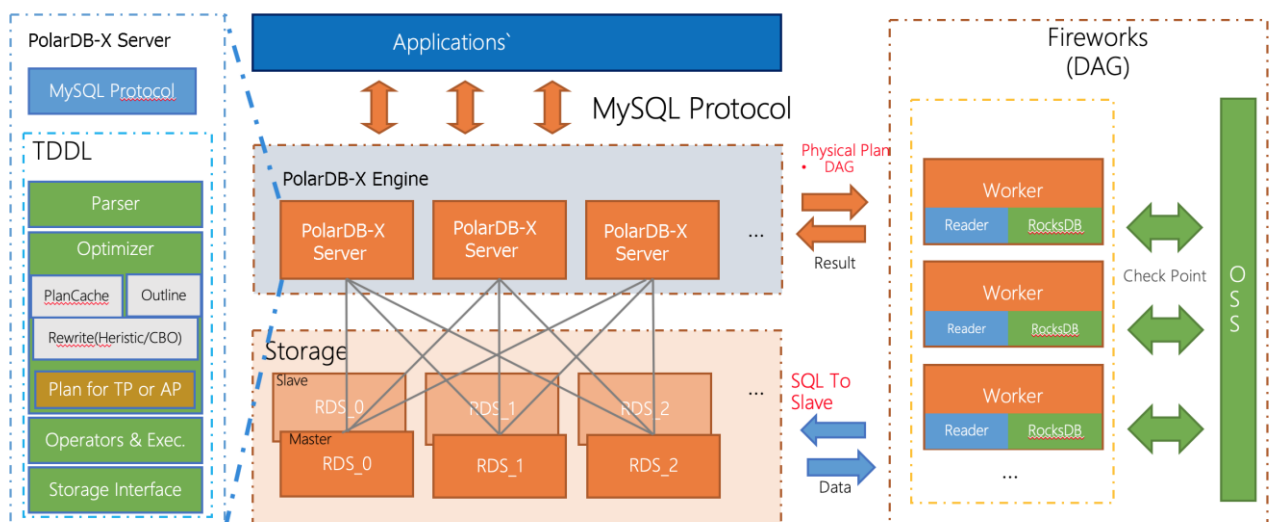
To process complex SQL statements in the online databases, PolarDB-X has expanded the Symmetric

Multi-Processing (SMP) and Massively Parallel Processing with Directed Acyclic Graph (MPP&DAG).

SMP is fully integrated into the PolarDB-X kernel, while MPP&DAG of PolarDB-Xbuilds a computing

cluster that dynamically obtains execution plans for distributed computing at runtime and improves the

computing capability by adding nodes.

Currently, thePolarDB-X instances that process data on multiple instances in parallel are provided for

businesses in the form of analytic read-only instances.
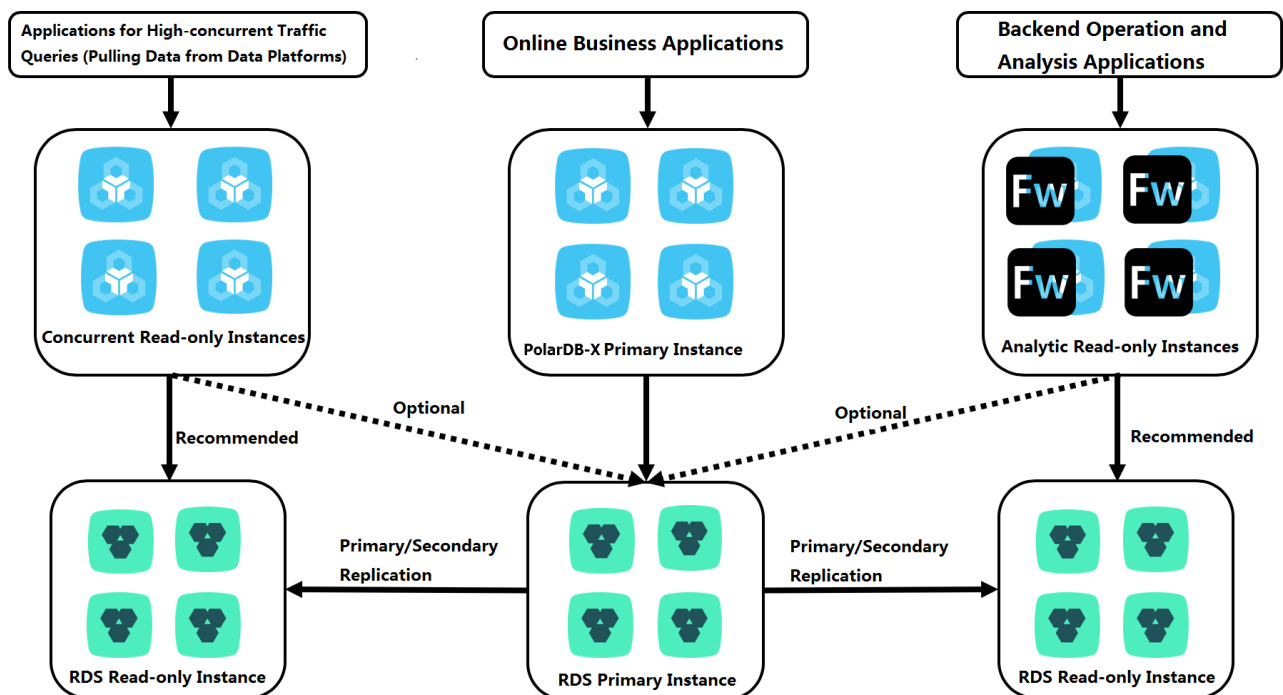
5.4.2. Distributed transactions

Distributed transactions use Two-Phase Commit (2PC) to ensure the atomicity and consistency of

transactions.

A 2PC transaction is divided into the PREPARE phase and the COMMIT phase.

- In the PREPARE phase, data nodes prepare all the resources required for committing

  transactions, such as locking and logging.

- In the COMMIT phase, data nodes commit transactions.

When you commit a distributed transaction, the PolarDB-X server, as a transaction manager, sends a

COMMIT request to each data node only after all data nodes (MySQL servers) have their resources

ready in PREPARE phase.
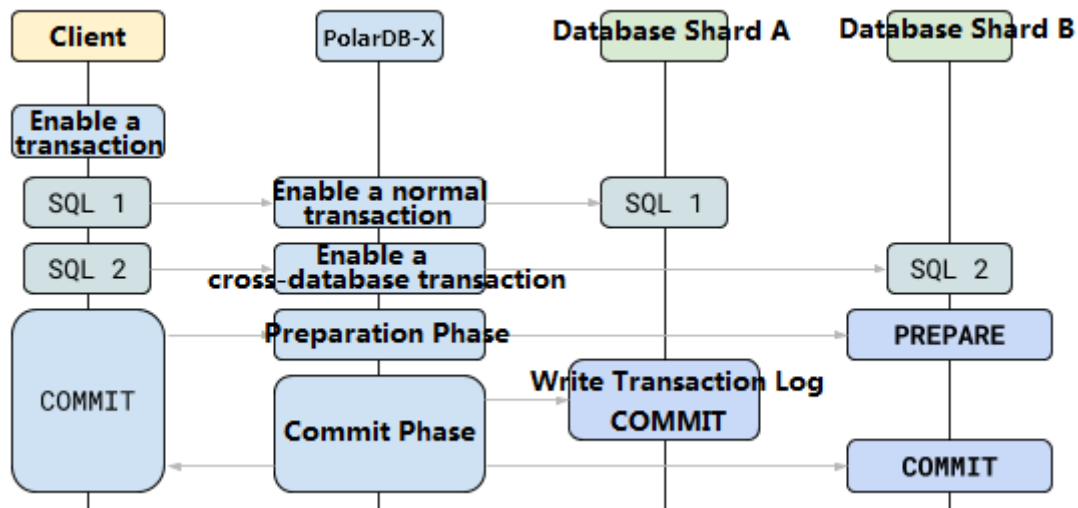
5.4.3. Smooth scale-out

When the underlying storage of the logical database reaches the physical bottleneck, for example,

when the remaining disk space is about 30%, you can smoothly scale it out to improve the

performance.

Smooth scale-out is an online horizontal expansion method. It smoothly migrates the original database

shards to the new ApsaraDB RDS for MySQL instances and increases the overall data storage capacity

by adding ApsaraDB RDS for MySQL instances, which reduces the pressure on each RDS instance to

process data.
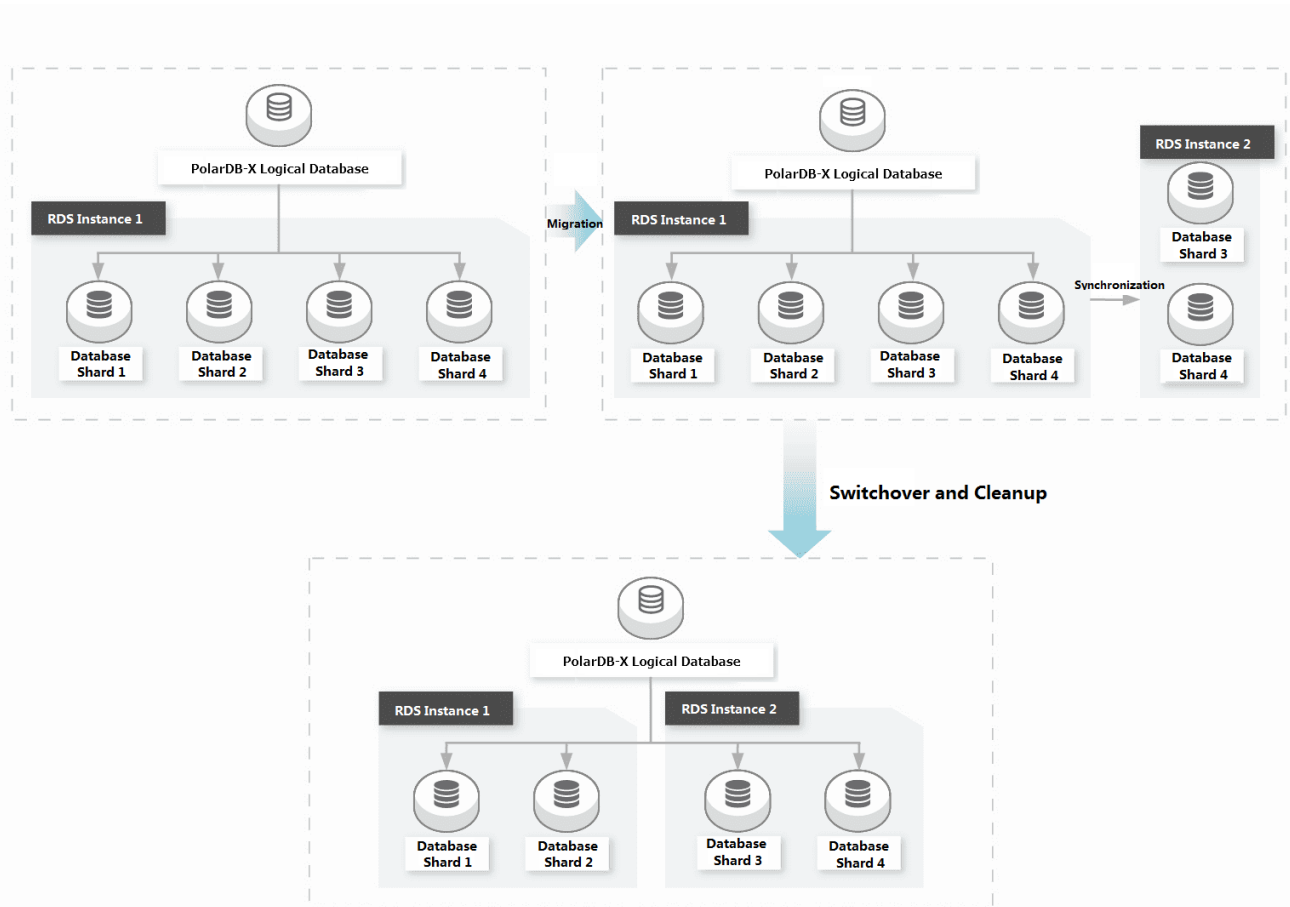
5.4.4. Read/write splitting

When a primary ApsaraDB RDS for MySQL instance is heavily loaded with many read requests, you

can use the read/write splitting function of PolarDB-X to distribute the read traffic, which reduces the

read pressure on the primary ApsaraDB RDS for MySQL instance.

The read/write splitting function of PolarDB-X is transparent to applications. The read traffic can be

distributed to the primary ApsaraDB RDS for MySQL instance and multiple ApsaraDB RDS for MySQL

read-only instances according to the read weight set in the PolarDB-X console, without changing any

code of the application. All the write traffic is distributed to the primary ApsaraDB RDS for MySQL

instance.

After read/write splitting is set, real-time strong consistency can be implemented when data is read from the primary ApsaraDB RDS for MySQL instance. Data on the read-only instances is replicated asynchronously from the primary ApsaraDB RDS for MySQLS instance, with a millisecond-level latency, therefore real-time strong consistency cannot be implemented when data is read from read-only ApsaraDB RDS for MySQL instances. For SQL statements which require real time and strong consistency for reading data, specify the primary ApsaraDB RDS for MySQL instance to execute these statements through hints of PolarDB-X.

## Read/write splitting in non-partition mode

In non-partition mode, PolarDB-X can implement read/write splitting without horizontal partitioning. When you create a PolarDB-X database in the PolarDB-X console, after you select an ApsaraDB RDS for MySQL instance, you can directly import a database in the instance to PolarDB-X for read/write splitting. In this case, you do not need to migrate data, but you also cannot perform horizontal partitioning on tables in the PolarDB-X database.

Support for transactions by read/write splitting

Read/write splitting is valid only for read requests (query requests) that are not in explicit transactions (transactions that need to be explicitly committed or rolled back). Write requests and read requests (including read-only transactions) in explicit transactions are executed in the primary instance and are not distributed to read-only instances.

- Common SQL statements for read requests include SELECT, SHOW, EXPLAIN, and DESCRIBE.

- Common SQL statements for write requests include INSERT, REPLACE, UPDATE, DELETE, and

  CALL.

Read/Write Splitting
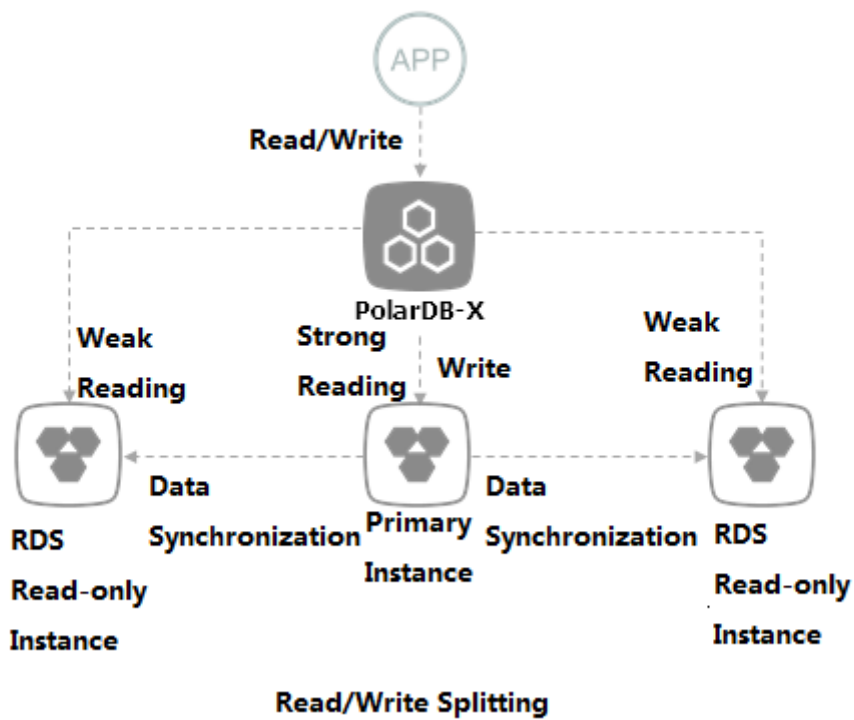
5.4.5. Global secondary index

Global secondary indexes of PolarDB-X allow users to add shard dimensions as needed and provides

globally unique constraints. Each global secondary index corresponds to an index table and uses XA

transactions to ensure strong data consistency between primary tables and index tables.
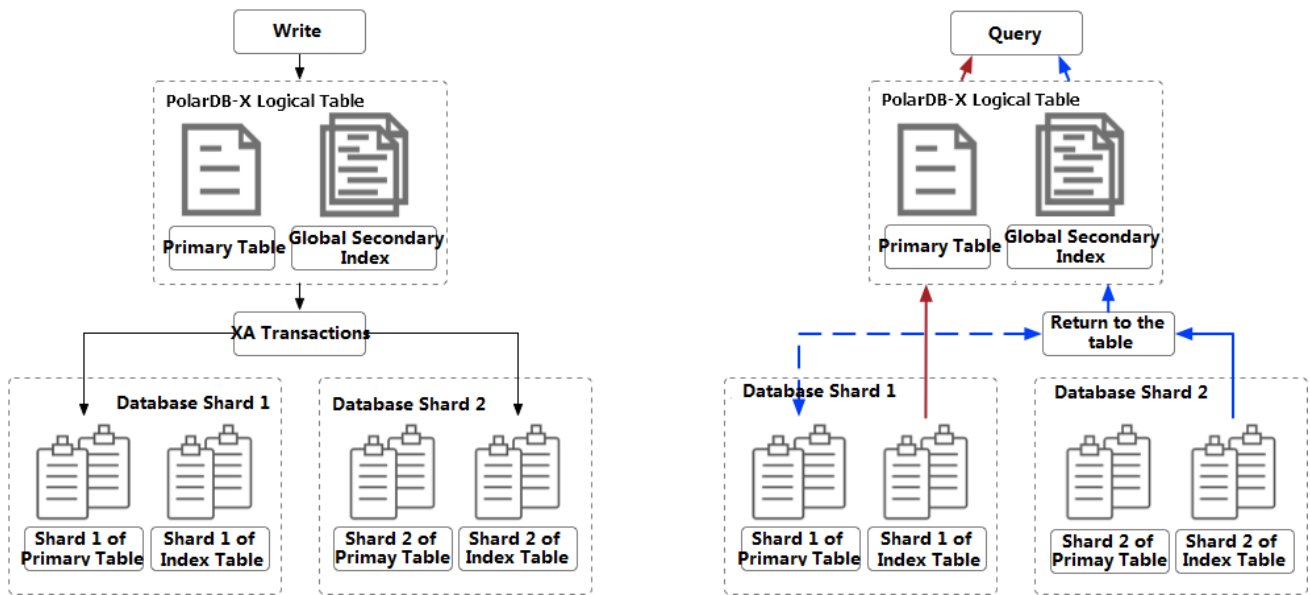
The global secondary indexes of PolarDB-X provide the following capabilities:

- Add dimensions for sharding.

- Support globally unique indexes.

- Provide XA transactions to ensure strong data consistency between primary tables and index

  tables.

- Support overwrite columns to reduce overheads from querying the primary table.

- Support Online Schema Change, so the primary table remains unlocked when a global secondary

  index is added.

- Uses hints to specify indexes to automatically determine whether to query the primary table.

FAQ

Q: What problems can global secondary indexes solve?

A: If the queried dimension is different from the dimension for sharding of a logical table, cross-shard

queries are initiated. As cross-shard queries increase, performance problems such as slow query and

connection pool exhaustion may occur. Global secondary indexes reduce cross-shard queries and eliminates performance bottlenecks by adding dimensions for sharding. When creating a global secondary index, you need to select a shard key that is different from that of the primary table.

Q: What is the relationship between a global secondary index and a local secondary index?

A:

- A local secondary index stores data rows and corresponding index rows on the same shard in a distributed database. In PolarDB-X, it specifically refers to a MySQL secondary index of a physical table.

- A global secondary index stores data rows and corresponding index rows on different shards, which is different from a local secondary index. A global secondary index quickly determines the data shards involved in the query.

- When PolarDB-X distributes queries to a single shard through a global secondary index, the local secondary index of the shard can improve the performance of the query within the shard.
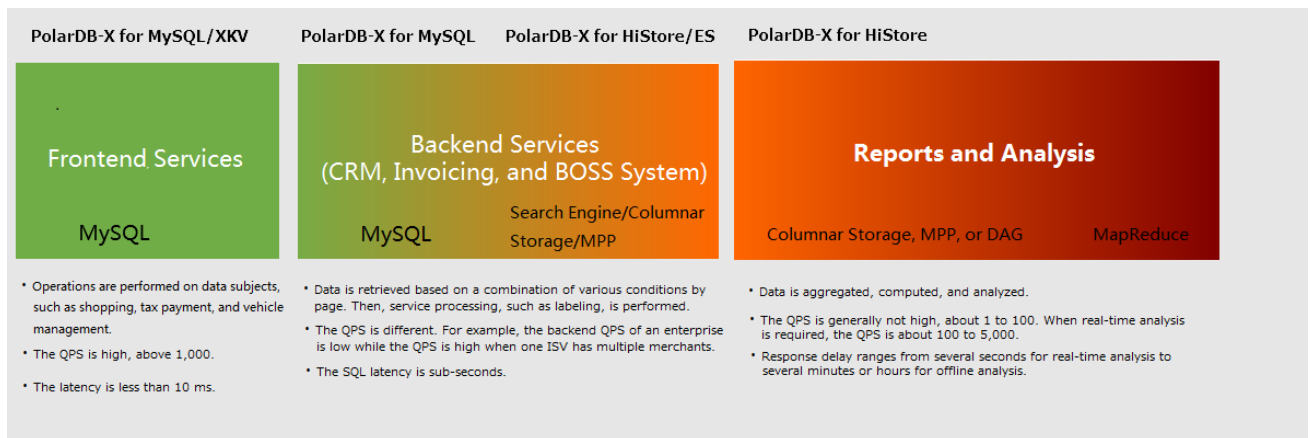
5.5. Scenarios

This topics describes the typical scenarios of PolarDB-X.

PolarDB-X is suitable for businesses that feature high concurrency and low latency in the frontend. It partitions data in specific business scenarios and provides distributed secondary indexes, enabling business databases to keep a high upper limit for queries per second (QPS).

PolarDB-X is trying to support Alibaba columnar databases to meet the needs of the huge-volume storage with low costs, efficient data aggregation, and ad hoc queries.

PolarDB-X scenarios

The following examples are business scenarios for your reference:

- Customer-oriented Internet applications to carry out the business for users (PolarDB-X for

  MySQL).

- Data businesses that feature high concurrency and low latency in the frontend, such as the bank

  and hospital counter businesses, Internet of Vehicles (IOV) data operations, tracing, and fuel

  consumption curves (PolarDB-X for MySQL).

- Storage and aggregation analysis of archived data that is unchangeable (including historical data),

  such as completed orders, logs, and operation and behavior records (PolarDB-X for HiStore).

5.6. Limits

This topic introduces the restrictions of using PolarDB-X.

Item	Limit
Table shard size	We recommend that a table shard contain a maximum of five million records.

Table shard quantity	Theoretically, the number of table shards in each database shard is not restricted, but depends on the hardware of the PolarDB-X server. .
Default database shard quantity for a single ApsaraDB RDS for MySQL instance	8, which cannot be changed.
Distributed JOIN	PolarDB-X supports most JOIN semantics, but also has some restrictions on complex JOIN semantics. For example, JOIN operations between large tables may result in performance or system unavailability due to the high cost and slow speed. Therefore, prevent it whenever possible.

5.7. Terms

This topic defines and analyzes the terms related to PolarDB-X.

Term	Description
PolarDB-X	PolarDB-X is developed by Alibaba. It is a distributed relational databases middleware that is highly compatible with the MySQL protocol and syntax.

PolarDB-XServer (PolarDB-X server node)	PolarDB-X Server is a core component of PolarDB-X. It provides the SQL statement parsing, optimization, routing, and result aggregation functions.
PolarDB-X instance	A PolarDB-X instance is a distributed database server cluster that consists of a group of PolarDB-X server nodes. Each server node is stateless and processes SQL requests.
Specifications of PolarDB-X instances	The specifications of PolarDB-X instances reflect the processing capability of PolarDB-X. Each type provides different CPU and memory resources. Instances with higher specifications provide higher processing capabilities. For example, in a standard PolarDB-X test scenario, an instance with an 8-core CPU and 16 GB of memory has twice the capability of an instance with a 4-core CPU and 8 GB of memory.
Instance upgrade and downgrade	PolarDB-X can adjust the processing capability by upgrading or downgrading instance specifications.
Horizontal partitioning (sharding)	The process that splits a single-instance database into multiple physical database shards, partitions and distributes table data from the single-instance into multiple physical table shards

	according to sharding rules, and then stores the table shards on different database shards.
Sharding rule	A rule used to partition a logical database table into multiple physical table shards during horizontal partitioning.
Shard key	A database field that generates sharding rules during horizontal partitioning.
Database shard	After the horizontal partitioning of PolarDB-X is complete, data in the logical database is stored in multiple physical storage instances. The physical database in each storage instance is a database shard.
Table shard	After the horizontal partitioning of PolarDB-X is complete, a physical data table in each database shard is called a table shard.
Logical SQL statement	The SQL statement sent by an application to PolarDB-X.
Physical SQL statement	The statement sent to ApsaraDB RDS for MySQL for execution after PolarDB-X parses a logical SQL statement.

Transparent read/write splitting	When a single storage node of PolarDB-X encounters an access bottleneck, you can add read-only instances to share the load on the primary instance.PolarDB-X You do not need to modify application code for the read/write splitting function, so it is called transparent read/write splitting.
Non-partition mode	PolarDB-X supports the extension of database service capabilities through transparent read/write splitting without horizontal partitioning. This is called the non-partition mode.
Smooth scale-out	PolarDB-X can scale out the database by adding storage instance nodes. Smooth scale-out does not affect access to original data.
Broadcast of small tables	PolarDB-X stores tables with small data volumes and infrequent updates in single table mode, which are called small tables. The solution which copies a small table to database shards related to it by JOIN statements through data synchronization to improve the JOIN efficiency, is called broadcast or replication of small tables.
Full table scan	In database partition mode, if no shard key is specified in the SQL statement, PolarDB-X executes the SQL statement on all

	table shards, merges the results and returns them. This process is called full table scan. To prevent impact on performance, we recommend that you do not perform a full table scan.
PolarDB-X sequence	A PolarDB-X sequence (a 64-digit number of the BIGINT data type in MySQL) aims to ensure that the data (for example, PRIMARY KEY and UNIQUE KEY) in the defined unique field is globally unique and in ordered increments.
PolarDB-X hint (PolarDB-X custom annotations)	A custom hint provided by PolarDB-X to specify certain special actions. It uses related syntax to control the SQL execution to optimize SQL statements.